Eötvös Loránd University
Institute of Mathematics

Erika Renáta Bérczi-Kovács

Network Coding Algorithms and Applications

Ph.D. thesis

Supervisor: András Frank
Professor, Doctor of Sciences

Advisor: Zoltán Király
Doctor of Philosophy

Doctoral School of Mathematics
Director: Miklós Laczkovich, Member of the Hungarian Academy of Sciences

Doctoral Program: Applied Mathematics
Director: György Michaletzky, Doctor of Sciences

Department of Operations Research, Eötvös Loránd University and
MTA-ELTE Egerváry Research Group on Combinatorial Optimization

Budapest, 2015

Contents

List of Figures

Notation

$D(V, A)$	A directed graph (shortly, digraph) on node set V with arc set A.
V^-	$V - s$ in a network with source node s.
D_X	The subgraph of D spanned by $X \subseteq V$.
$\varrho_D(X)$	The number of arcs entering $X \subseteq V$.
$\Delta_D^{in}(X)$	The set of edges entering $X \subseteq V$.
$\delta_D(X)$	The number of arcs leaving $X \subseteq V$.
$\Delta_D^{out}(X)$	The set of edges leaving $X \subseteq V$.
$\lambda(u, v)$	The maximum number of arc-disjoint directed paths from u to v in a digraph.
$\overline{u}v$-cut X	$X \subseteq V$ such that $u \notin X$ and $v \in X$.
L	The set of consecutive pairs of arcs in a digraph.
$\mathbb{F}_q$	The finite field of size q.
$\mathbb{F}_q^k$	The k-dimensional vector space over $\mathbb{F}_q$.
$\mathbf{e}_i$	The ith unit vector in a vector space $\mathbb{F}_q^k$.
$\langle S \rangle$	The linear subspace spanned by vectors in S.

Acknowledgement

First of all, I am greatly indebted to András Frank for giving me the opportunity to study and work among the members of the EGRES group, and for serving as an example of a researcher and teacher. I am also very grateful to him for providing a wide range of research experiences, both in theory and applications, in Hungary and abroad as well. These occasions highly influenced my interest in several topics.

I cannot be grateful enough to Zoltán Király for his time and unlimited patience during our work. He encouraged me to work on some topics when I had given up. Undoubtedly, without him this thesis could neither have been started nor finished. Chapters 2 and 4 are joint work with him.

I also thank the support of the Communication Networks Laboratory, and the opportunity to participate on its workshops, which raised my interest in computer science related problems. It was during these events that I met Péter Babarczi, Gábor Rétvári, and János Tapolcai. I really enjoyed the work with them, and the wonderful atmosphere they created during brainstorming. I became familiar with the topic of Chapter 5 on these occasions.

Working on practical problems in network coding was one of the most interesting experiences during my studies, I am very thankful to Frank Fitzek, Daniel Lucani and Morten Pedersen for kindly hosting me in Aalborg, it was a pleasure and lot of fun working with them on Chapter 3.

I thank the EGRES group and my colleagues at ELTE for the inspiring atmosphere they created, especially Katalin Vesztergombi and Tibor Jordán for the supporting words and useful advices.

I am thankful to Martin Skutella and the COGA group, especially Jannik Matuschke and Ronald Koch for their hospitality during my visit at COGA. I would like to express my thanks to Martin Grötschel and the Zuse Institute Berlin, and also Siemens AG for their scholarship program, I was very glad to work on real applications of combinatorial optimization during my visits in

Berlin and Münich. I also thank Ambros Gleixner and Stefan Heinz for their help, and for introducing me to SCIP. I am grateful to Tamás Szőnyi for the opportunity to participate in the First European Training School in Network Coding.

I am indebted to my teachers for guiding me in the world of Mathematics, especially Tünde Fazakas, Márta Táborné Vincze, Katalin Zsuga Jánosné and Lajos Deli. I try to achieve the standards they set on every class I teach.

I would like to thank all my friends, especially Fruzsi and Tomi for their support, kindness and that I could always share ups and downs with them.

I am forever grateful to my parents for the loving family background they provided, and the example they showed.

Finally, I thank my husband, Kristóf Bérczi so many things... like correcting the biography of this thesis or staying up late with me during work. And for his guitar play, he is the most amusing beginner I know. And last but not least, I thank our soon-to-be-born baby for giving a seriously hard deadline to finish this thesis.

Chapter 1

Introduction

1.1 Network coding: a new tool for information transmission

Network coding is a fairly new area on the boundary of information theory and graph theory, where in contrast to a classical source-terminal encoding-decoding information transmission scheme, intermediate nodes of the network are also enabled to perform coding.

Transportation problems can be classified according to the nature of the transported objects, leading to different combinatorial optimization problems. For example, when physical objects are to be sent in a network, transportations have to be carried out through different paths, leading to the problem family of path packings. Whereas if the object is a piece of information e.g. an e-mail that has to be sent to a group of people, the sender can make use of the fact that *information can be copied* at internal nodes of the network, leading to Steiner tree/arborescence packing problems.

Network coding can be regarded as a second step in capturing the nature of information transmission by letting nodes not only to copy but also *to transmit some function of the incoming messages*. These functions are called coding functions. Usually the messages sent are considered as members of some finite field, and coding functions are restricted to linear combinations.

What can be the advantage of such an operation? Properly chosen coding functions may increase throughput, security or reliability of a network [47].

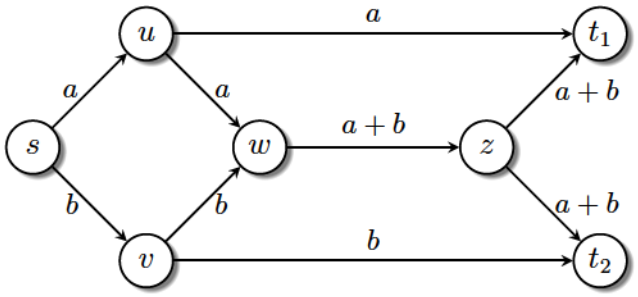

Figure 1.1: A network code on the so-called Butterfly network, sending two messages from s to t_1 and t_2 simultaneously.

The well-known Butterfly network is one of the smallest acyclic examples demonstrating the throughput advantage of network coding (see Figure 1.1). In this example, node s transmits two messages (a and b) to two nodes (t_1 and t_2). Each arc has capacity one. Although there exists two arc-disjoint paths from s to both terminals, it is still impossible to send both messages to both terminals simultaneously using simple routing, because there exist no two arc-disjoint s-rooted Steiner-arborescences spanning t_1 and t_2.

In their seminal paper from 2000 [1], Ahlswede, Cai, Li and Yeung proposed the application of network coding for information transmission. They introduced a very general framework for possible coding functions for all graphs (as we will see, later results are mostly restricted to acyclic graphs). They considered the problem of sending k messages from a source to a set of terminals simultaneously (multicasting), and proved a min-max-type theorem for the solvability of the problem by showing that the existence of k arc-disjoint paths from the source to each of the terminals is a necessary and sufficient condition.

This result is a natural analogue of Edmonds' theorem about packing spanning arborescences [11]. From theoretical point of view its importance was giving a new tool for achieving information theoretical throughput bounds for a wider class of information transmission problems than simple routing.

1.2 Previous work

Although Ahlswede et al. in [1] investigated general graphs, possibly containing cycles, most later results consider acyclic digraphs only, mostly because of the applicability of linear network codes in this special case. Through the thesis, we will also be restricted to the acyclic case.

1.2.1 Definitions

First, we introduce some graph theoretical notations. Assume that a digraph $D = (V, A)$ is given with $|V| = n$ nodes and $|A| = m$ arcs. Let $\lambda(u, v)$ denote the maximum number of arc-disjoint paths from u to v in D. For a set $X \subseteq V$, let $\varrho(X)$ and $\delta(X)$ denote number of arcs entering and leaving X, respectively. Similarly, let $\Delta^{in}(X)$ and $\Delta^{out}(X)$ denote the *set* of arcs entering and leaving X, respectively. A **topological order of the nodes** is an ordering $v_1, \ldots, v_n$ such that for every arc $v_i v_j \in A$ we have $i < j$. It is easy to see that a digraph has a topological order if and only if it is acyclic. Given a topological order of the nodes, we can similarly the **topological order of the arcs** as an ordering $a_1, a_2, \ldots, a_m$ such that if $i < j$, then the tail of arc a_i is no later than the tail of arc a_j in the topological order.

Definition 1.2.1. A **network** is an acyclic directed graph $D = (V, A)$ with a single **source** node $s \in V$ and a set of **terminal/receiver** nodes (assumed to be sinks) $T \subseteq V - s$. Nodes in $V \setminus (T + s)$ are called **internal** nodes. For simplicity we assume that every node is reachable from the source.

Let $\mathbb{F}_q$ be a finite field of size q and let $\mathbb{F}_q^k$ denote the k-dimensional vector space over $\mathbb{F}_q$, where k is the number of messages to be sent in the network. Let $\mathbf{e}_i$ denote the ith unit vector. For a set $S \subseteq \mathbb{F}_q^k$ of vectors, we denote the linear subspace spanned by S by $\langle S \rangle$.

A data stream is divided into k messages of equal size. At each time slot t, a set of messages $\mathbf{M}(t) = \{M_1(t), M_2(t), \ldots M_k(t)\}$ is generated at the source,

message $M_i(t)$ belonging to the ith message and represented by an element of $\mathbb{F}_q$. The task is to multicast $\mathbf{M} = (M_1, M_2, \dots M_k)$ from s to T.

Definition 1.2.2. Given a network, let $L \subseteq A \times A$ be the set of consecutive pairs of arcs: $L = \{(wu, uv) |\ w, u, v \in V,\ wu, uv \in A\}$. For the sake of shortness, members of L are called **pairs**. A **network code** is described by two functions $(\alpha, \mathbf{c})$ referring to local and global coefficient functions, respectively. The **local coefficient function** of a network code is a function $\alpha : L \to \mathbb{F}_q$. The **global coefficient function** of a network code is a function $\mathbf{c} : A \to \mathbb{F}_q^k$ such that

$$\mathbf{c}(uv) = \sum_{wu \in A} \alpha(wu, uv)\mathbf{c}(wu)$$

for every arc $uv \in A, u \neq s$.

Sometimes for the sake of shortness we refer to local and global coefficient functions as local and global coefficients, respectively.

From a network code we get actual transmissions on an arc uv by the scalar product $\mathbf{M} \cdot \mathbf{c}(uv)$. For example, in Figure 1.1 the global coefficients on arcs su, ut_1 and uw are all $(1, 0)$, but $\mathbf{c}(wz) = (1, 1)$ and local coefficient $\alpha(uw, wz) = \alpha(vw, wz) = 1$.

Remark 1.2.3. Assume that global coefficients $\mathbf{c}(sv) \in \mathbb{F}_q^k$ are given on every arc in $\Delta^{out}(s)$ together with a local coefficient function α. Note that from the acyclic property of the graph we get that these values uniquely determine global coefficients on every arc in the graph.

Definition 1.2.4. For a network code $(\alpha, \mathbf{c})$, a node v **can decode** (or receives) message M_i, if $\mathbf{e}_i \in \langle \{\mathbf{c}(uv) | uv \in A\} \rangle$. A network code is **feasible** for terminal set T, if for every terminal node $t \in T$ the dimension of $\langle \mathbf{c}(vt) | vt \in A \rangle$ is k.

Note that simple routing can be regarded as a special case of network coding, where for each arc uv, $\mathbf{c}(uv) = \mathbf{e}_i$ for some $1 \leq i \leq k$.

We remark here that if a node v can decode message i by the above definition, then it really can decode message M_i, as it gets all scalar products $\mathbf{c}(uv) \cdot \mathbf{M}$ and $\mathbf{e}_i \in \langle \mathbf{c}, v \rangle$, so it can easily calculate $M_i = \mathbf{e}_i \cdot \mathbf{M}$.

Definition 1.2.5. Given a positive integer k and a network with digraph $D = (V, A)$, source s, terminal set T, the **network coding problem** is to decide whether there exists a feasible network code in the network over some finite field.

1.2.2 Algorithm for network code construction

The fundamental result in [1] proved that the cut bound is sufficient for a wide class of network coding problems. However, because of the general class of possible coding functions, these solutions would be hard to implement in practice. In their paper [32], Li, Yeung and Cai showed that in fact the family of linear coding functions is satisfactory for achieving optimal throughput.

Theorem 1.2.1 (Li, Yeung, Cai [32])**.** *Assume a network coding problem is given. There exists a feasible network code over some finite field if and only if $\lambda(s,t) \geq k$ for every terminal $t \in T$.*

In Theorem 1.2.1 the required field size had to be very large. Later, Koetter and Médard radically improved the lower bound for the sufficient field size.

Theorem 1.2.2 (Koetter, Médard [27])**.** *Assume a network coding problem is given. There exists a feasible network code over any finite field $\mathbb{F}_q$ with $q > |T|k$ if and only if $\lambda(s,t) \geq k$ for every terminal $t \in T$.*

Finally, Jaggi et al. gave a field size bound of $|T|$ and a polynomial time algorithm for network code construction [21]. Here we present their result altogether with its proof, as their algorithm will be generalized in several chapters.

Theorem 1.2.3 (Jaggi et al. [21])**.** *Assume a network coding problem is given such that $\lambda(s,t) \geq k$ for every terminal $t \in T$. Then there exists a feasible*

network code over every field $\mathbb{F}_q$ with $q > |T|$. Moreover, such a feasible network code can be constructed in polynomial time.

Proof. Without loss of generality we may assume that $\delta(s) = k$, otherwise an extra source s' can be added to the digraph with k parallel $s's$ arcs. A network code on the extended digraph can be naturally mapped to one on D.

Now let $a_1, a_2, \ldots, a_m$ be a topological order of the arcs and for $1 \leq i \leq m$, let $A_i := \{a_1, \ldots, a_i\}$. Then $A_k = \{a_1, \ldots, a_k\} = \Delta^{out}(s)$. Similarly, let $L_i \subseteq L$ denote the set of those pairs for which both arcs are in A_i. For every terminal $t \in T$, let us fix k arc-disjoint st-paths $P_1^t, \ldots, P_k^t$. From the assumption of the theorem these paths exist. For every fixed path P_i^t and arc set A_j, we have that $P_i^t \cap A_j$ is a subpath of P_i^t starting from s. Let $P_i^t[j]$ denote the last arc on this subpath. Initially we define $\mathbf{c}(a_i) := \mathbf{e}_i$ for $1 \leq i \leq k$. Then the network code is determined in the topological order of the arcs, maintaining the following property:

$$\langle \mathbf{c}(P_1^t[j]), \ldots \mathbf{c}(P_k^t[j]) \rangle = \mathbb{F}_q^k \text{ for every terminal } t \text{ and } k \leq j \leq m.$$

This clearly holds for $j = k$. For $j > k$, assume that local and global coefficients are defined on L_{j-1} and A_{j-1}, respectively. Let $a_j := uv$ and let $\Delta^{in}(u) := \{w_1u, w_2u, \ldots, w_{\varrho(u)u}\}$, finally let T_{uv} be the set of terminals that use arc uv on a path P_x^t and for these terminals let w_tu be the arc before uv on path P_x^t. We define local coefficients $\alpha(w_iu, uv)$ in the order $i = 1, 2, \ldots, \varrho(u)$. Initially all local coefficients $\alpha(w_iu, uv)$ are set to zero. Let us denote $\sum_{\ell=1}^{i} \mathbf{c}(w_\ell u)\alpha(w_\ell u, uv) := \mathbf{c}_i(uv)$. When fixing a value of a local coefficient $\alpha(w_iu, uv)$, we maintain that for those terminals in T_{uv} for which $w_tu \in \{w_1u, \ldots, w_iu\}$,

$$\langle \{\mathbf{c}(P_1^t[j-1]), \ldots \mathbf{c}(P_k^t[j-1])\} - \mathbf{c}(w_tu) + \mathbf{c}_i(uv) \rangle = \mathbb{F}_q^k.$$

Assume that local coefficients $\alpha(w_1u, uv), \alpha(w_2u, uv), \ldots, \alpha(w_{i-1}u, uv)$ are already fixed and the assumption holds. In order to prove the existence of a

proper local coefficient $\alpha(w_i u, uv)$, some key observations from linear algebra are needed. Their proof can be found for example in [21].

Claim 1.2.6. *Let $\{\mathbf{v}_1, \ldots, \mathbf{v}_{k-1}\}$ be a $k-1$ dimensional subspace of $\mathbb{F}_q^k$. Then there exists a vector $\mathbf{y} \in \mathbb{F}_q^k$ such that for every vector $\mathbf{x} \in \mathbb{F}_q^k$, the set $\{\mathbf{v}_1, \ldots, \mathbf{v}_{k-1}\} + \mathbf{x}$ is a basis of $\mathbb{F}_q^k$ if and only if $\mathbf{y} \cdot \mathbf{x} \neq 0$. Such a vector can be found in polynomial time.*

□

During the proof of Theorem 1.2.3 such a vector for a $k-1$ element set of independent vectors is called a control vector of the set. For each terminal t in T_{uv}, consider sets $\{\mathbf{c}(P_1^t[j-1]), \ldots \mathbf{c}(P_k^t[j-1])\} - \mathbf{c}(P_x^t[j-1])$ and fix a control vector $\mathbf{y}_t$ for each set.

Claim 1.2.7. *Let vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{F}_q^k$ form a basis of $\mathbb{F}_q^k$ and let $\mathbf{v} \in \mathbb{F}_q^k$. Then*

i) there is at most one value $\alpha \in \mathbb{F}_q$ not *satisfying that the set $\{\mathbf{v}_1' = \mathbf{v}_1 + \alpha\mathbf{v}\} \cup \{\mathbf{v}_i\}_{i=2}^k$ is also a basis,*

ii) there is at most one value $\beta \in \mathbb{F}_q$ not *satisfying that the set $\{\mathbf{v}_1' = \beta\mathbf{v}_1 + \mathbf{v}\} \cup \{\mathbf{v}_i\}_{i=2}^k$ is also a basis.*

□

When choosing a proper value for $\alpha(w_i u, uv)$, we apply the claim simultaneously for each terminal t if $P_x^t[j-1] \in \{w_1 u, w_2 u, \ldots, w_i u\}$. If $P_x^t[j-1] = w_i u$, we apply Case ii) with $\mathbf{v}_1 = \mathbf{c}(w_i u)$ and $\mathbf{v} = \mathbf{c}_{i-1}(uv)$ and $\{\mathbf{v}_2, \ldots, \mathbf{v}_k\} = \{\mathbf{c}(P_1^t[j-1]), \ldots \mathbf{c}(P_k^t[j-1])\} - \mathbf{c}(P_x^t[j-1])$, whereas if $P_x^t[j-1] \in \{w_1 u, w_2 u, \ldots, w_{i-1} u\}$, we apply Case i) with $\mathbf{v} = \mathbf{c}(w_i u)$ and $\mathbf{v}_1 = \mathbf{c}_{i-1}(uv)$ and $\{\mathbf{v}_2, \ldots, \mathbf{v}_k\} = \{\mathbf{c}(P_1^t[j-1]), \ldots \mathbf{c}(P_k^t[j-1])\} - \mathbf{c}(P_x^t[j-1])$. Since $|T_{uv}| \leq |T| < q$, we get that there is at least one value for $\alpha(w_i u, uv)$ such that all basis transformations result in a basis, which can be found by using the vectors $\mathbf{y}_t$. For $i = \varrho(u)$ we get exactly the inductive assumption for j, which proves the theorem. □

1.3 Applications of network coding

Over the only fifteen-year-old history of network coding, the method has turned out to be useful in numerous transmission scenarios. In this section we present some of these applications which motivated the topics of this thesis. Most applications presented here are generalizations of the network coding problem defined in the previous section. Each will be presented in a more detailed form in the following chapters. We have to mention, though, that there are several other possible applications of network coding, such as network security or distributed storage systems, that are not contained in the scope of this thesis.

1.3.1 Increased throughput

The Butterfly network (see Figure 1.1) showed an example when network coding outperforms simple routing. There has been tremendous work on measuring the possible advantage of network coding compared to simple routing in different scenarios [35, 36, 37]. Jaggi et al. in [21] showed, that the advantage of network coding over simple routing can be $\Omega(\log n)$ in acyclic networks. In case of undirected graphs, Li, Li and Lau proved that the advantage can be bounded by 2 [35]. For the multiple unicast transmission problem in undirected networks Li and Li conjecture an advantage of 1 [34].

In Chapter 2, we consider a variation of the network coding problem where receivers' demands may be different, but have a certain laminar structure. This model is applied for the multi-layered video streaming problem, and algorithms are proposed to increase throughput compared to routing. We give an optimal polynomial time algorithm for two layers when the goal is to send the base layer to every user, and within this constraint to maximize the number of users receiving two layers. For the case of three or more layers we show NP-hardness of the problem. Also, we show NP-hardness for the case of two layers when the goal is to maximize the total number of transmitted useful layers. We also propose a heuristic for three layers and give experimental

comparison between the best known heuristic due to Kim et al. [22] and our approach [24]. Finally, we show an algorithm for calculating the expected performance of some randomized heuristics.

1.3.2 Fast coupon collection

Probably the most promising directions towards real-life applications of network coding are the ones capturing its advantage in coupon collector-like scenarios.

The classical coupon collector problem is the following: there are k different types of coupons. A little girl would like to collect all of them. In each round, she gets a coupon chosen randomly with uniform distribution on the coupon types. What is the expected value of the number of rounds she needs to wait until she has at least one coupon from each type?

Such scenarios appear for example on wireless broadcasting channels with failures. Assume that a group of users would like to receive the same set of k messages from a wireless broadcasting source, but each user has a fairly high probability of packet loss. Without network coding, the only possible strategy of the source is to transmit one of the messages in each time slot. From a user's point of view the transmission is equivalent to coupon collection. In this case, the expected number of transmissions is $O(k \log k)$.

With network coding however, expected value of required transmissions can be decreased radically. Suppose that in each round, some random linear combination of the messages are sent to the users. If the applied field size is large enough, the expected number of rounds is nearly the expected number of rounds with k successful transmissions, which is the best one can expect.

A similar scenario is considered in Chapter 3 with the application to multi-layered video streams. We introduce a network coding scheme for the problem that takes user diversity into account. We present with both simulations and estimations proving that the expected completion time can be reduced with the proposed method [28].

1.3.3 Modelling interference in wireless networks

In contrast to wired networks, which can usually be modelled by a graph or digraph, wireless networks cannot be substituted with such a unified model, mainly because of their divers properties regarding failure probability and interference. A recent approach of Avestimehr, Diggavi and Tse [3] proposed a model for Gaussian relay networks, which estimated failure probability with deterministic values, and modelled interference with sums over finite fields. This new approach opened the door for determining capacity of these networks by applying techniques of combinatorial optimization. As it turned out, the proposed finite field sum model for interference is actually a special case of a network coding problem called deterministic network coding or network code completion problem (NCCP), when values on a subset of the local coefficients are previously fixed.

In Chapter 4, we give both randomized and deterministic algorithms for maximum throughput-achieving network code construction for the NCCP in the multicast case. We also introduce a related problem called fixable pairs, investigating when a certain subset of coding coefficients in the linear combination functions can be fixed to arbitrary non-zero values such that the network code can always be completed to achieve maximum throughput. We give a sufficient condition for a set of coding coefficients to be fixable. For both problems we present applications in different wireless and heterogenous network models [25, 26].

1.3.4 Failure protection

The last topic, detailed in Chapter 5, considers the problem of protection of failures in a network. When designing a protection scheme, two opposite challenges have to be considered: on the one hand, we expect the scheme to react to a failure as soon as possible, but on the other hand low capacity occupation is preferred. For example, if instant recovery is required for sending k messages from a source to a receiver, then with simple routing all messages

have to be sent on two arc-disjoint paths, thus occupying twice the capacity of the original message set. With network coding however, the problem can be solved with only $k+1$ arc-disjoint paths $P_1, \ldots, P_k, P_{k+1}$, by sending the original messages on paths $P_1, \ldots, P_k$ and a linear combination of them on path P_{k+1}. In Chapter 5, we mention applications of network coding for the multicast, multiple failure case and for the unicast capacitated problem. We show efficient algorithms and lower field size bounds for network code construction [8], and present some negative results for the capacity case of the problem [5].

Chapter 2

Multi-layer video streaming

2.1 Introduction

The appearance of new devices (smartphones, tablets, etc.) has highly increased user diversity in communication networks. As a consequence, when broadcasting a video stream, users may have very different quality demands depending on their resolution capabilities.

Multi-resolution code (MRC) is one successful way to handle this diversity, encoding data into a base layer and one or more refinement layers [12, 40]. Receivers can request cumulative layers, and the decoding of a higher layer always requires the correct reception of all lower layers (including the base layer). The multi-layer multicast problem is to multicast as many valuable layers to as many receivers as possible.

In a multi-layered streaming setup, network coding was shown to be a successful tool for increasing throughput compared to simple routing [22]. In their simple heuristic [22], Kim et al. give a network coding scheme based on restricting the set of layers that may be encoded at certain nodes.

This chapter proposes algorithms for the multi-layered video streaming problem. We give an optimal polynomial time algorithm for two layers when the goal is to send the base layer to every user, and within this constraint to maximize the number of users receiving two layers. For the case of three or more layers we show NP-hardness of the problem. Also, we show NP-hardness for the case of two layers when the goal is to maximize the total number of transmitted useful layers. We also propose a heuristic for three layers and

give experimental comparison between the best known heuristic due to Kim et al. [22] and our approach. Finally we show an algorithm for calculating the expected performance of some randomized heuristics.

The rest of the chapter is organized as follows: Section 2.2 contains the problem formulation and some definitions. In Section 2.3 we prove NP-hardness for some special cases of the problem. In Section 2.4, the notion of feasible height function is introduced, and a sufficient condition for simultaneously satisfiable receiver demands is given. In Section 2.5 we give an optimal algorithm for two layers. In Section 2.6 we present a heuristic for three layers, and give some numerical results of experimental comparison. In Section 2.7 we give an algorithm for calculating the expected performance of some randomized layered network coding heuristics such as those in [22].

2.2 Problem Formulation

Definition 2.2.1. In multi-resolution coding, for $i > j$ we say that layer i is **higher** than layer j, and layer j is **lower** than layer i. The **height** of a network code on an arc uv is the highest layer with non-zero coefficient on that arc. For example, the first unit vector has height one and so on, $\mathbf{e}_i$ has height i, and vector $(1, 0, 1, 0)$ has height 3. The height of $\mathbf{c}$ is denoted by $h_c : A \to \mathbb{N}$.

A layer i is **valuable** for a node only if all lower layers can also be decoded at that node, i.e., for every $j \leq i$ message M_j is decodable. The **performance** of a network code at a node v is the index of the highest valuable layer for v. The performance function of $\mathbf{c}$ is denoted by $p_c : V \to \{0, 1, \ldots, k\}$, where $p(v) = 0$ denotes that layer 1 is not decodable at v.

A **demand** is a sequence of mutually disjoint subsets of V^- denoted by $\tau = (T_1, T_2, \ldots, T_k)$. The set of **receiver nodes** is the union of these request sets, denoted by $T = T_1 \cup T_2 \cup \ldots \cup T_k$. The nodes in T_i **request** the first i layers. Given a demand τ, we can define a **demand function** $d_\tau : V \to \{0, 1, \ldots, k\}$ on the nodes a straightforward way setting $d_\tau(v) = i$ if $v \in T_i$, and $d_\tau(v) = 0$

if $v \in V \setminus T$.

A network code is **feasible** for demand τ, if every receiver node $t \in T_i$ can decode M_j for all i and $j \leq i$, that is, $p_c(v) \geq d_\tau(v)$ for all $v \in V$. If there exists a feasible network code for a demand, the demand is called **satisfiable**. The **multi-layered network coding problem** is to decide whether there exists a feasible network code for a given demand.

2.3 Complexity Results

In this section we prove NP-hardness of some special cases of the multi-layered network coding problem. Lehman and Lehman [31] showed NP-hardness for a more general network coding problem, where receivers may demand any subset of the messages, and there can be multiple sources, accessing disjoint subsets of the information demanded by the receivers. Here we need and prove NP-hardness for a special case of this problem, when there is only one sink in the graph, and demands of the receivers have a layered structure as defined in the previous section 2.2.1.

Theorem 2.3.1 (B-K, Király [24]). *Given a directed acyclic graph D and a demand with three layers $\tau = (T_1, \emptyset, T_3)$, it is NP-hard to decide, whether there exists a feasible network code for τ.*

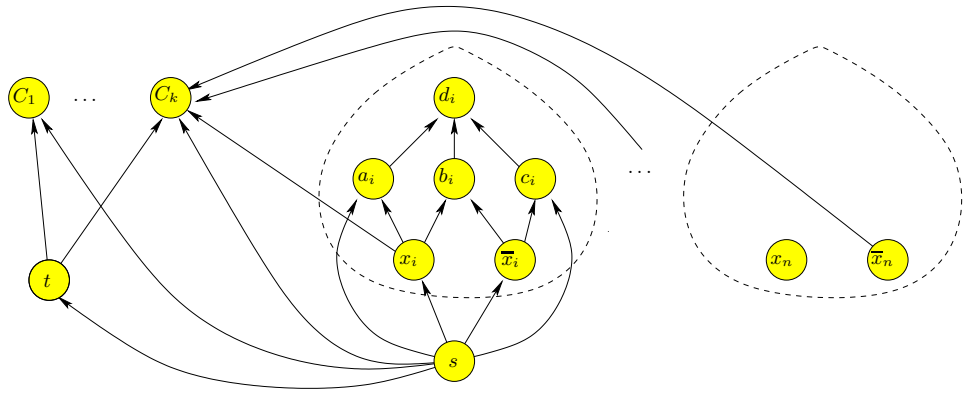

Figure 2.1: Reduction of 3-SAT to demand $\tau = (T_1, \emptyset, T_3)$.

Proof. We reduce 3-SAT to this problem. Let $S = (X, CL)$ be a 3-SAT instance, where $X = \{x_1, \ldots, x_n\}$ and $CL = \{C_1, \ldots, C_m\}$ denote the set of variables and clauses, respectively. We define a network coding problem on a digraph D corresponding to this instance. First we create special nodes s, t with an arc st, and put t into T_1. For each variable x_i we add six nodes with eleven arcs (see Figure 2.1), so that $a_i, b_i, c_i \in T_1$ and $d_i \in T_3$. Nodes x_i and $\overline{x}_i$ correspond to literals. For each clause C_j we add a node C_j, arcs sC_j and tC_j and arcs from every node corresponding to literals of C_j. Each C_j is put into T_3. We prove that this network coding problem has a feasible solution over some finite field if and only if S can be satisfied. Suppose the above defined network coding problem has a feasible network code $\mathbf{c}$. Since $t \in T_1$, $h_c(st) = 1$ and for all C_j $\;h_c(tC_j) = 1$. Moreover, the arc sC_j can transmit any message from s, hence C_j can decode all three layers if and only if at least one additional arc entering C_j has height greater than one. Note that such an arc can only leave a node corresponding to a literal in C_j.

Claim 2.3.1. *If the network coding problem has a feasible network code* $\mathbf{c}$*, then for every variable* x_i*, the code* $\mathbf{c}$ *has height one on at least one of the arcs* sx_i *and* $s\overline{x}_i$*.*

Proof. Let us assume indirectly that neither $\mathbf{c}(sx_i)$ nor $\mathbf{c}(s\overline{x}_i)$ have height one. Since a_i, b_i, c_i must be able to decode the first layer, $\mathbf{c}(sa_i) \in \langle \mathbf{e}_1, \mathbf{c}(sx_i) \rangle$, and $\mathbf{c}(sc_i) \in \langle \mathbf{e}_1, \mathbf{c}(s\overline{x}_i) \rangle$, and $\mathbf{e}_1 \in \langle \mathbf{c}(sx_i), \mathbf{c}(s\overline{x}_i) \rangle$. Hence we have

$$\dim\langle \mathbf{c}(sa_i), \mathbf{c}(sc_i), \mathbf{c}(sx_i), \mathbf{c}(s\overline{x}_i) \rangle = \dim\langle \mathbf{c}(sx_i), \mathbf{c}(s\overline{x}_i) \rangle \leq 2,$$

that is, these four vectors cannot span a 3-dimensional space to transmit three layers to d_i. □

From the claim we can transform a solution of the network coding problem into an assignment of S by assigning value 'true' to a literal l if the height of $\mathbf{c}(sl)$ is at least two. Note that if for a variable x_i both sx_i and $s\overline{x}_i$ have height one, we can choose the value of x_i arbitrarily to get a satisfying assignment.

Similarly we can get a feasible network code $\mathbf{c}$ for the network coding problem from a truth assignment of S over any field. The corresponding $\mathbf{c}(e)$ vectors are the following. Let $\mathbf{c}(st) = (1,0,0)$, $\mathbf{c}(sC_j) = (1,1,1)$, and for any node u with only one incoming arc wu, all outgoing arcs carry $\mathbf{c}(wu)$. If x_i is true, then $\mathbf{c}(sa_i) = (0,1,0)$, $\mathbf{c}(sx_i) = (1,1,0)$, $\mathbf{c}(s\overline{x}_i) = (1,0,0)$, $\mathbf{c}(sc_i) = (1,1,1)$, $\mathbf{c}(a_id_i) = (0,1,0)$, $\mathbf{c}(b_id_i) = (1,0,0)$, $\mathbf{c}(c_id_i) = (1,1,1)$, and the code can be constructed symmetrically if x_i is false. It is easy to check that this $\mathbf{c}$ is indeed a feasible network code. □

For the general case with $k \geq 3$ layers we easily get the similar result by adding $k-3$ new sC_i and sd_j arcs for each i, j.

Corollary 2.3.2. *For $k \geq 3$ layers and demand $\tau = (T_1, \emptyset, \ldots, \emptyset, T_k)$, it is NP-hard to decide whether there exists a feasible network code for τ.*

Theorem 2.3.2 (B-K, Király [24])**.** *Given a directed acyclic graph $D = (V, A)$ and a demand $\tau = (T_1, T_2)$ it is NP-hard to find a maximal cardinality subset T_1' of T_1, so that for $\tau' = (T_1', T_2)$ there exists a feasible network code.*

Proof. We prove the theorem by reducing the Vertex Cover problem. Let $G = (W, E)$ denote an instance of this problem. For every vertex $w \in W$ we add a receiver $t_w \in T_1$ with an arc st_w, while for every edge $uv \in E$ we add a receiver $t_{uv} \in T_2$ with arcs t_ut_{uv} and t_vt_{uv}. For a given network code $\mathbf{c}$, a receiver node t_w can decode the first layer if and only if the height of the code on st_w is one. A receiver node t_{uv} can decode both layers, if on at least one entering arc the code has height two. Let $T_1' \subseteq T_1$ denote the set of nodes $t_w, w \in W$ for which the arc st_w has height 2. It is easy to conclude that if the code is feasible for demand $(T_1 \setminus T_1', T_2)$ then T_1' is a vertex cover. Conversely, from a vertex cover $T_0 \subseteq W$ we get a feasible network code for demand $\tau_0 = (T_1 \setminus T_0, T_2)$ with height 2 on arcs incident to nodes in T_0, because for a fieldsize large enough ($q \geq |W|$), all arcs with height two entering the same receiver can be chosen to be independent. □

As a minimal mixed (vertices and edges) cover of the edges may be supposed to contain only vertices, we also get the following.

Corollary 2.3.3. *Given a network with demand $\tau = (T_1, T_2)$ and a number K, it is NP-hard to decide whether there exists a network code satisfying at least K requests.*

Remark 2.3.4. In a recent work Widmer et al. [44] considered another version of the multi-layer network coding problem, when internal nodes of the network cannot perform decoding. Precisely, the height of an outgoing arc of an internal node can only be a height of an incoming arc of that node. In the case of two layers they showed that the proof of Theorem 2.3.2 can be adapted for proving NP-hardness of maximizing the number of receivers being able to decode two layers if every receiver must get at least one layer. As we will see in Section 2.5, this problem is solvable in polynomial time if decoding at internal nodes is allowed.

2.4 Tools for feasible network code construction

In [22] Kim et al. gave a simple randomized network coding algorithm for the multi-layered video streaming problem. In their approach a function $h : V \to \{0, 1, \ldots, k\}$ is determined, and then a randomized linear network code $\mathbf{c}$ is sent in the network such that for each arc $uv \in A$, the highest layer with non-zero coefficient in $\mathbf{c}(uv)$ is at most $h(v)$. Their algorithm ensures that the first layer can be decoded at each receiver with high probability, and some receivers may be able to decode more layers.

In this section we give some (non-randomized) algorithms that are also based on restricting the highest layer with non-zero coefficient, but in our approach restrictions may differ for arcs entering the same node. In order to describe our algorithms, some further layer-related notions are needed.

Definition 2.4.1. A function $f : A \to \{0, 1, \ldots, k\}$ is a **height function** if there exists a finite field $\mathbb{F}_q$ and a linear network code $\mathbf{c}$ over $\mathbb{F}_q$ with $h_c = f$. Similarly we can define when a function $g : V \to \{0, 1, \ldots, k\}$ is a **performance function**, i.e., if there exists a linear network code $\mathbf{c}$ over

$\mathbb{F}_q$ with $p_c = g$. We say that functions $f : A \to \{0, 1, \ldots, k\}$ and $g : V \to \{0, 1, \ldots, k\}$ form a **height-performance-pair** if there exists a network code c with $h_c = f$ and $p_c = g$. Given a function $f : A \to \{0, 1, \ldots, k\}$, a function $g : V \to \{0, 1, \ldots, k\}$ is called a **realizable extension** of f, if they form a height-performance-pair. A height function f is **feasible** for a demand τ if it has a realizable extension g such that $g \geq d_\tau$.

2.4.1 Sufficient condition for feasible height functions

Our algorithms for feasible network code construction for a demand τ will always first find a function $f : A \to \{0, 1, \ldots, k\}$ and then a realizable extension g such that $g(v) \geq d_\tau$.

In this subsection we give a sufficient condition for a function f to be a height function (see Corollary 2.4.7). As we will see, this condition is also necessary for two layers, leading to a characterization of that case. For three layers it is applied for a new heuristic with better performance than earlier approaches.

In this section we assume that the reader is familiar with the classical algorithm of Jaggi et al. [21]. In their algorithm, they construct a feasible network code for a demand $\tau = (\emptyset, \ldots, \emptyset, T_k)$ by fixing k arc-disjoint paths to every receiver and constructing the network code on the arcs one by one, in the topological order of their tails. We say that an arc a is **processed** during the algorithm, if the network code $\mathbf{c}(a)$ is defined. Jaggi et al. maintain that for every receiver, the span of the codes on the last processed arcs on the fixed k paths remain the whole k-dimensional vector space. Their algorithm can be easily generalized for multi-layer demands.

Definition 2.4.2. For a function $f : A \to \{0, 1, \ldots, k\}$, a path P with arcs $a_1, a_2, \ldots, a_r$ is called **monotone**, if $f(a_1) \leq f(a_2) \leq \ldots \leq f(a_r)$. We define for such a monotone path $\min(P) = f(a_1)$ and $\max(P) = f(a_r)$.

Definition 2.4.3. Let a node $v \in V^-$, a function $f : A \to \{0, 1, \ldots, k\}$ and a function $g : V \to \{0, 1, \ldots, k\}$ be given. An i**-fan** of v consists of i

pairwise arc-disjoint non-trivial (i.e., containing at least one arc) monotone paths $P_1, \ldots, P_i$ ending at v, where for all $j \leq i$ we have $j \leq \min(P_j) \leq \max(P_j) \leq i$, and P_j begins at a node v_j with $g(v_j) \geq \min(P_j)$.

Definition 2.4.4. If a function $f : A \to \{0, 1, \ldots, k\}$ and a function $g : V \to \{0, 1, \ldots, k\}$ is given such a way that

i, for every node v with $g(v) > 0$ there exists a $g(v)$-fan of v,

ii, for every arc vw, either $f(vw) \leq g(v)$, or there exists an incoming arc uv with $f(uv) = f(vw)$,

then g is called a **fan-extension** of f. Let us call an arc uv **free** if $f(uv) \leq g(u)$. Note that every starting arc of a path in a fan is free.

Theorem 2.4.1 (B-K, Király [24])**.** *A fan-extension g of a function f is also a realizable extension of f.*

Proof. If a node can decode the first i layers then it can also send any linear combination of these layers.

Claim 2.4.5. *If v has an i-fan then it also has an i-fan with exactly one free arc on each path.*

Proof. Let a' be a free arc on a path P_j of a fan such that it is not the first arc a. Since P is monotone, $j \leq f(a) \leq f(a')$, hence the fan resulting from replacing P_j by the subpath P'_j starting from a' to v is also an i-fan of v. □

Let us fix such a fan for every node v with $g(v) > 0$. First we define the network code $\mathbf{c}$ on arcs covered by at least one fan. Let L denote the maximum number of fans an arc is covered by. Our algorithm constructs a network code over any finite field F_q with $q > L$. Note that since $|V| > L$, $q > |V|$ is always sufficient. We modify the algorithm of Jaggi et al. [21] the following way: on free arcs of a fan we construct the network code in increasing order of the f values on the arcs. Since the paths in a fan satisfy that $\min(P_j) \geq j$ and $q > L$, we can define the network code $\mathbf{c}$ so that for

every fan, $\dim\langle \mathbf{c}(a_1), \ldots, \mathbf{c}(a_j)\rangle = j$ for all $1 \leq j \leq i$, where a_j is the first arc on path P_j. On non-free arcs we define the network code in a topological order of their tails. When constructing the network code on a non-free arc $uv, u \neq s$, we maintain that for every i-fan which contains uv, the span of codes on the last processed arcs on the i paths remain the i-dimensional subspace of the first i layers. We use the following lemma to prove that this is possible.

Lemma 2.4.6 (Jaggi et al. [21])**.** *Let $n \leq q$. Consider pairs $(\mathbf{x}_i, \mathbf{y}_i) \in \mathbb{F}_q^k \times \mathbb{F}_q^k$ with $\mathbf{x}_i \cdot \mathbf{y}_i \neq 0$ for $1 \leq i \leq n$. There exists a linear combination $\mathbf{b}$ of vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ such that $\mathbf{b} \cdot \mathbf{y}_i \neq 0$ for $1 \leq i \leq n$.*

If vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ span the subspace of the first n layers, then for every $\mathbf{v}_i, 1 \leq i \leq n$ there is a vector $\mathbf{y}_i$ in this subspace with $\mathbf{v}_j \cdot \mathbf{y}_i = 0, i \neq j$ and $\mathbf{v}_j \cdot \mathbf{y}_j \neq 0$. We call $\mathbf{y}_j$ a **control vector** of $\mathbf{v}_j$. Let $F_1, \ldots, F_\ell$ denote the set of fans containing uv. Consider first F_1, suppose it is an i-fan. Let $P_1, \ldots P_i$ denote the paths of fan F_1. For $j = 1, \ldots, i$ let a_j denote the last processed arc of P_j, and let $\mathbf{v}_j = \mathbf{c}(a_j)$. After determining control vectors $\mathbf{z}_1, \ldots, \mathbf{z}_i$, let $\mathbf{x}_1 = \mathbf{v}_p$ and $\mathbf{y}_1 = \mathbf{z}_p$, where path P_p is the one that uses arc uv. Clearly arc a_p enters u, so $\mathbf{x}_1 = \mathbf{c}(a_p)$. For the other fans we similarly define $\mathbf{x}_2, \mathbf{y}_2, \ldots, \mathbf{x}_\ell, \mathbf{y}_\ell$. Let wu be an entering arc with $f(wu) = f(uv)$. Since uv is a non-free arc, such an arc exists. Define $\mathbf{x}_{\ell+1} = \mathbf{c}(wu)$ and $\mathbf{y}_{\ell+1} = \mathbf{e}_{f(u,v)}$. Now apply Lemma 2.4.6, it gives the linear combination $\mathbf{b}$, define $\mathbf{c}(uv) = \mathbf{b}$. The height of $\mathbf{c}(uv)$ will be at most the height of arcs $w_i u$, hence it remains under $f(uv)$, because all P_j's are monotone. As $\mathbf{b} \cdot \mathbf{y}_{\ell+1} = \mathbf{b} \cdot \mathbf{e}_{f(uv)} \neq 0$, the height of uv is exactly $f(uv)$. Finally, for arcs not covered by any fan we can choose $\mathbf{c}$ arbitrarily within the height constraint. Because of property ii, in Definition 2.4.4, this can also be done in the topological order of the tails of these arcs. □

Corollary 2.4.7. *If a function $f : A \to \{0, 1, \ldots, k\}$ has a fan-extension then f is a height function.*

2.4.2 Maximal fan-extensions

In this subsection we prove a key property of fan-extensions.

Theorem 2.4.2 (B-K, Király [24]). *If a function f has a fan-extension, then it has a unique maximal fan-extension g^* such that $g^*(v) \geq g(v)$ for every fan-extension g of f and every node v.*

First we start with a very important, though straightforward observation.

Proposition 2.4.8. *Given a fan-extension g of a function f such that there exists an i-fan to a node v with $i > g(v)$, setting $g(v)$ to i is also a fan-extension of f.*

Proof of Theorem 2.4.2 . Let g^+ be a fan-extension for which $\sum_{v \in V} g^+(v)$ is maximum and assume indirectly that there exists another fan-extension g' and a node v for which $g'(v) > g^+(v)$. We can assume that v is the first such node in a topological order. From Proposition 2.4.8, increasing g^+ on v to $i = g'(v)$ would also give a fan-extension, because the i-fan of v is also an i-fan for g^+. □

Theorem 2.4.3 (B-K, Király [24]). *The maximal fan-extension of a function f can be determined algorithmically.*

Proof. From Proposition 2.4.8, we can calculate the maximal fan-extension in a topological order of the nodes. Assume that g is defined for any node before a node $v \in V^-$ in that order. For a given value $0 \leq i \leq k$, let $D_{v,i} = (V', A')$ denote the following auxiliary graph of D: we delete all arcs with f value greater than i. We add i extra nodes to the digraph: $t_1, \ldots, t_i$ with $2i - 1$ extra arcs: $st_j,\ 1 \leq j \leq i$ and $t_j t_{j+1},\ 1 \leq j \leq i - 1$. For every node u before v in the topological order we change the tail of every outgoing arc uw from u to $t_{f(uw)}$ if $g(u) \geq f(uw)$.

Lemma 2.4.9 (B-K, Király [24]). *There exists an i-fan to $v \in V$ if and only if $\lambda_{D_{v,i}}(s, v) = i$.*

Proof. Note that a monotone path P to v in D, with exactly one free arc, corresponds to a path in $D_{v,i}$ starting from $t_{\min(P)}$ and vice versa. Hence an i-fan corresponds to i paths in $D_{v,i}$, each starting from a node t_j for some j. Suppose indirectly that $\lambda_{D_{v,i}}(s,v) < i$, that is, there exists an $\overline{s}v$ set $X \subseteq V'$ with $\varrho(X) < i$. Since $\lambda_{D_{v,i}}(s,t_i) = i$, $t_i \notin X$. Let j denote the greatest integer for which $t_j \in X$. Since for an i-fan at least $i-j$ paths in the fan have value at least $j+1$, paths in $D_{v,i}$ corresponding to paths of the fan enter X on at least $i-j$ arcs. Also, there are j paths to t_j in $D_{v,i}$ using arcs between s and $t_1,\ldots,t_j$ only, which are disjoint from the arcs of the fan. Hence there are at least i arcs entering X, contradicting the assumption.

To prove the other direction let $P_1, P_2, \ldots, P_i$ be i arc-disjoint sv paths in $D_{v,i}$. Note that $\{t_1,\ldots,t_i\}$ is a cut set in $D_{v,i}$ hence a path P_j must go through at least one of them. Since $\varrho(\{t_1,\ldots,t_j\}) = j$, at least $i-j+1$ paths go through the set $\{t_{j+1},\ldots,t_i\}$, which correspond to paths in D with value at least $j+1$. □

The maximal possible value of $g(v)$ is the maximal i for which there exist an i-fan of v. Once g is determined for every node, we can easily check property ii, in Definition 2.4.4 for f and g. □

Lemma 2.4.9 shows that the existence of a fan is equivalent with a connectivity requirement in an auxiliary graph.

Corollary 2.4.10. *Given a function $f : A \to \{0,1,\ldots,k\}$ and a demand τ, we can check algorithmically whether f has a fan-extension g such that $g \geq d_\tau$ by calculating the maximal fan-extension g^* and comparing it to d_τ.*

2.5 Characterizing feasible height functions for two layers

For two layers ($k = 2$) the feasible height functions can be characterized. For a node $v \in V$, let $\lambda(s,v)$ denote the maximal number of arc-disjoint paths

from s to v. A demand is **proper**, if $\lambda(s, t_i) \geq i$ for all i and all $t_i \in T_i$. Being a proper demand is a natural necessary condition for a demand to have a feasible network code, however, not always sufficient.

Theorem 2.5.1 (B-K, Király [24]). *A function $f : A \to \{1, 2\}$ is a height function, feasible for a proper demand $\tau = (T_1, T_2)$, if and only if for all arcs $uv \in A$, $u \neq s$*

1. *if $f(uv) = 2$, then $\exists wu \in A : f(wu) = 2$,*
2. *if $f(uv) = 1$, then either $\exists wu \in A : f(wu) = 1$, or $\lambda(s, u) \geq 2$, and moreover*
3. *for any receiver $t \in T_1$ with $\lambda(s, t) = 1$, there is a 1-valued arc entering t, and*
4. *for any $t \in T_2$ there is a 2-valued arc entering t.*

Proof. Let $U \subseteq V^-$ denote the set of special non-receiver nodes, where a node u is special, if all entering arcs are 2-valued, but it has a 1-valued outgoing arc (by Property 2, we know that $\lambda(s, u) \geq 2$). The set of receiver nodes $t \in T$ for which $\lambda(s, t) = 1$ is denoted by T_1'. As τ is proper, for each node in $T_2' = U \cup T \setminus T_1'$ there exist two arc disjoint paths from s, hence, for receiver set T_2' there exists a network code $\mathbf{c}$ feasible for demand $\tau_2 = (\emptyset, T_2')$. If the field size q is greater than $|T_2'|$, the code can be chosen to have height two on every arc, that is, the coefficient of $\mathbf{e}_2$ is nonzero [21]. In order to be feasible for the original demand $\tau = (T_1', T_2')$, we modify $\mathbf{c}$ the following way: for every arc uv with $f(uv) = 1$ we set $\mathbf{c}(uv) = (1, 0)$.

We are left to prove that $\mathbf{c}$ remains a network code, and becomes feasible for demand τ.

The span of the incoming vectors can only change at nodes which have only 1-valued incoming arcs, but in this case it has also only 1-valued outgoing arcs, so the network code has the linear combination property (note that in special nodes the span of the incoming vectors remains two-dimensional). Using Properties 3 and 4, the code clearly becomes feasible for demand τ. □

Corollary 2.5.1. *For two layers, a function $f : A \to \{0, 1, \ldots, k\}$ is a height function feasible for a demand τ if and only if it has a fan-extension g with $g(v) \geq d_\tau$ for all v.*

Proof. We use the notations of the previous proof and define the extension g to be 2 on T_2' and 1 on T_1' and zero everywhere else. It is easy to see that for a receiver node t, if it is in T_1', there is a path from another terminal node containing 1-valued arcs only, that is, there exists a 1-fan to that node. If t is in T_2', either there are two edge disjoint paths of 2-valued arcs starting from receiver nodes both in T_2' or there is a path of 1-valued arcs from a node in T_1' and a path of 2-valued arcs from a node in T_2'. Both cases give a 2-fan for t. □

2.5.1 Optimal algorithm for two layers

In this subsection we show that given the condition that all receiver nodes have to be able to decode the first layer, there is a unique maximal set of nodes X in the graph such that demand $\tau' = (T \setminus X, X)$ is satisfiable. We will give an algorithm for finding this maximal set, as well as constructing a feasible network code.

Definition 2.5.2. For nodes u, v in a digraph $D = (V, A)$, a set $X \subseteq V$ is an $\overline{u}v$ **set** if $v \in X$ but $u \notin X$. For a set X of nodes let $\varrho(X)$ denote the number of entering arcs of X. A set X not containing s, and having $\varrho(X) = i$ is called an i**-set**.

Note that by Menger's theorem, $\lambda(s, v)$ equals the minimum of $\varrho(X)$, where X is an $\overline{s}v$ set.

Proposition 2.5.3. *Let $v \in V^-$, $\lambda(s, v) = i$, and X, Y two i-sets with $v \in X \cap Y$. Then $X \cup Y$ is also an i-set.*

Proof. As $\varrho(X \cup Y) + \varrho(X \cap Y) \leq \varrho(X) + \varrho(Y)$, and $\varrho(X \cup Y),\ \varrho(X \cap Y) \geq i$, the claim follows. □

From the claim we get that for every vertex $v \in V^-$ there is a unique maximal $\lambda(s,v)$-set containing v.

Given a proper demand $\tau = (T_1, T_2)$, the following algorithm gives a feasible height function for $\tau' = (T_1, T_2')$ where T_2' is the unique maximal subset of T_2, such that a feasible network code for τ' exists. As a by-product, it also decides whether demand τ is satisfiable or not. Having this height function, one can easily get a feasible network code for τ' by the lines of the previous subsection. We remark that this code will also be feasible for $\tau'' = (T_1 \cup (T_2 \setminus T_2'), T_2')$, in other words every receiver will get at least the base layer. We will also prove, that any fieldsize $q > |T_1| + |T_2|$ will be enough for this network code.

Let $\{Z_i\}$ be the maximal 1-sets which contain at least one node from T. Let $I(Z_i)$ denote the set of arcs with head or tail in Z_i.

Claim 2.5.4. *The sets Z_i are pairwise disjoint and so are the sets $I(Z_i)$.*

Let Z denote the set of nodes not reachable from s in $D' = (V, A \setminus \bigcup_i I(Z_i))$. It is obvious that if every receiver in T can decode the first layer, then no receiver in Z can decode two layers. Let $T_2' = T_2 \setminus Z$. For an arc $uv \in A$, let $f(uv)$ be the following. If $uv \in I(Z)$, then $f(uv) = 1$, otherwise let $f(uv) = 2$.

Theorem 2.5.2 (B-K, Király [24])**.** *Function f is realizable for $\tau'' = (T_1 \cup (T_2 \setminus T_2'), T_2')$. In addition, any finite field of size $q > |T|$ can be chosen for the network code (where $T = T_1 \cup T_2$).*

Proof. By the definition of Z, it is clear that Constraint 1 of Theorem 2.5.1 is fulfilled. Suppose that $f(uv) = 1$ for an arc with $u \neq s$ and there are no 1-valued arcs entering u. We need to prove that $\lambda(s,u) \geq 2$.

Suppose that this is not the case, thus there is an $\overline{su}$ set $X \subset V$ with $\varrho(X) = 1$. Since $uv \in I(Z)$ but none of the arcs entering u is in $I(Z)$, it follows that $v \in Z$ and $u \notin Z$. Hence $v \in Z_i$ for some i, but then $X \cup Z_i$ would be a subset with in-degree one, contradicting the maximality of Z_i.

For the second statement, using the proof of Theorem 2.5.1, it is enough to show that the size of the set U of special non-receiver nodes defined there is not greater than the number of terminals that have demand one in τ''. We claim moreover that $|U| \leq |T \cap Z|$. Every $u \in U$ is a tail of an arc entering some Z_i, and for every Z_i there is only one entering arc. Since each of the pairwise disjoint sets Z_i contains at least one terminal from $T \cap Z$, we are done. □

We note that this algorithm has a more-or-less obvious implementation in time $O(|A|)$ using BFS. We do not detail it here, because a more general algorithm given in the next section will also do the job.

2.6 Three layers

2.6.1 Heuristics for 3 layers

In this subsection we give a new network coding algorithm for three layers. We prove that the algorithm sends the first layer to every receiver and within this constraint, the unique maximal set of receivers gets at least two layers, while some receivers may get three layers. Because of its properties we call our heuristic 2-Max.

Step 1 Let W_1 denote the union of maximal 1-sets which contain at least one node from T. In Section 2.4 it was proved that if all receivers get the first layer, a receiver v cannot get more than one layer if and only if it is cut by W_1 from s, that is, if there is no directed path from s to v in $V \setminus W_1$. Let $\overline{W}_1 \supseteq W_1$ denote the set of nodes cut from s by W_1. We set $T_1 = T \cap \overline{W}_1$. We define a set of **pseudo receivers** U which contains nodes not in $\overline{W}_1$ but having an outgoing arc entering $\overline{W}_1$.

Step 2 Similarly to the first case, let W_2 denote the maximal 2-sets which contain a receiver or a pseudo receiver. Let $\overline{W}_2 \subseteq V \setminus \overline{W}_1$ denote the set of nodes only reachable from s through $\overline{W}_1 \cup W_2$. We set $T_2 = (T \cup U) \cap \overline{W}_2$.

Note that for determining sets $\overline{W}_1$ and $\overline{W}_2$ we can use the distributed algorithm presented in Subsection 2.6.2.

Step 3 We define a function $f : A \to \{0, 1, \ldots, k\}$ on D which is 1 on $I(\overline{W}_1)$, 2 on $I(\overline{W}_2) \setminus I(\overline{W}_1)$ and 3 otherwise. Let $T^* = U \cup T$. We proceed on the nodes of $T^* \setminus T_1$ in a fixed a topological order and decrease f on some arcs from 3 to 2. Let v denote the next node to be processed. We take a cost function $c : A \to \{0, 1\}$ which is 1 on 3-valued arcs and 0 everywhere else. Then we take the set of nodes $X \subseteq T_3$ reachable from s on 3-valued arcs and increase c to 1000 on an s-arborescence (a directed tree in which every node except s has in-degree 1) of 3-valued arcs spanning X. Since $v \notin \overline{W}_1$, there are two arc-disjoint paths P_1 and P_2 from $T^* \cup \{s\}$ to v so that P_2 does not start in T_1. Moreover, it can be assumed that the inner nodes of these paths do not intersect T^*.

Case I There are two edge-disjoint paths from $T^* \cup \{s\}$ to v, such that both avoid T_1. Let us take a minimum cost pair of paths $P_1 \cup P_2$ described above according to the cost function c. Then we decrease f on the 3-valued arcs of P_1 and P_2.

Case II No such pair exists. We take a minimum cost $P_1 \cup P_2$ from $T^* \cup \{s\}$ to v according to the cost function c. Then we decrease f on the 3-valued arcs of P_1 and P_2.

Step 4 Finally, we check in the topological order of the nodes, whether every 3-valued outgoing arc has a 3-valued predecessor, and if not, we decrease its value to 2.

Theorem 2.6.1 (B-K, Király [24])**.** *The function f constructed has a realizable extension for demand $\tau = (T_1, T_2', T_3')$ for which $T_2 \subseteq T_2'$ and $(T_2' \cup T_3') \supseteq T \setminus T_1$. Heuristic 2-Max sends at least one layer to each receiver and within this constraint it sends at least two layers to the maximum number of receivers.*

2.6.2 A connectivity algorithm for determining maximal 1-sets and 2-sets

Goals: we are going to give a distributed, linear time algorithm for the following problems:

- Determine $\lambda(s, v)$ for all v, but if it is ≥ 3 then only this fact should be detected.
- For each v with $\lambda(s, v) = 1$ determine the incoming arc of the unique maximal 1-set containing v.
- For each v with $\lambda(s, v) = 2$ determine the incoming arcs of the unique maximal 2-set containing v.

We assume that $*$ is a special symbol which differs from all arcs.

During the algorithm each node v (except s) waits until it hears messages along all incoming arcs, then it calculates $\lambda(s, v)$, and the 3 messages $m_1(v), m_2(v), m_3(v)$ it will send along all outgoing arcs.

The algorithm starts with s sending $m_1(s) := m_2(s) := m_3(s) := *$ along all outgoing arcs.

We need to describe the algorithm for an arbitrary node $v \in V^-$. First v waits until hearing the messages on the set of incoming arcs denoted by $IN(v) = \{a_1, \ldots, a_r\}$. When on an arc a_i it hears a $*$, it replaces it by a_i. Let the messages arrived (after these replacements) on arc a_i be m_1^i, m_2^i, m_3^i. Then v examines the set $M_1(v) = \{m_1^i\}_{i=1}^r$. If $|M_1(v)| = 1$ then v sets $\lambda(s, v) := 1$ and $m_1(v) := m_2(v) := m_3(v) := m_1^1$, otherwise it sets $m_1(v) := *$.

Next v examines the set $M_2(v) = \bigcup_{i=1}^r \{m_2^i, m_3^i\}$. If $|M_2(v)| = 2$ then it sets $\{m_2(v), m_3(v)\} = M_2(v)$, and if $\lambda(s, v)$ was not set to 1 before, it sets it to 2.

Let us call an entering arc a_j **important** for v, if $m_1^j \notin \bigcup_{1 \leq i \leq r, i \neq j} \{m_2^i, m_3^i\}$, and let I_v denote the set of important arcs for v. If $|M_2(v)| > 2$, then v next examines the set $M_2'(v) = \bigcup_{i \in I_v} \{m_2^i, m_3^i\} \cup \bigcup_{i \notin I_v} \{m_1^i\}$, and if $|M_2'(v)| = 2$, then it makes the same steps with $M_2'(v)$ as described before with $M_2(v)$.

Finally, if both $|M_2'(v)|$ and $|M_2(v)|$ are greater than 2 and $\lambda(s,v)$ was not set to 1, v examines $M_1(v)$ again, and if $|M_1(v)| \leq 2$, then it sets $\{m_2(v), m_3(v)\} = M_1(v)$, and it sets $\lambda(s,v)$ to 2.

If they were not set before, let $m_2(v) := m_3(v) := *$ and $\lambda(s,v) = 3$.

An sv **cut** is a set of arcs, which intersects every sv path.

Claim 2.6.1. *Let $v \in V^-$. Each of the sets $M_1(v)$, $M_2(v)$ and $M_2'(v)$, whenever defined, contains an sv cut.*

Proof. For an arc a, let us call an arc set an a-arc-cut if it intersects every directed path from s ending with a. Note that the arc a itself is an a-arc-cut, and the union of arc-cuts for all the entering arcs of a node v form an sv cut. Also, for an arc uv, an su cut forms a uv-arc-cut. To prove the claim, inductively we can assume, that on an arc uv either $m_1(uv) = *$ or $m_1(uv)$ is an su cut, and also either set $\{m_2(uv), m_3(uv)\} = \{*\}$ or is an su cut. In all cases, after the replacement, node v hears along arc uv an m_1 that forms a uv-arc-cut and m_2, m_3 that form also a uv-arc-cut, proving the claim. □

Theorem 2.6.2 (B-K, Király [24])**.** *For every node $v \in V^-$, the algorithm correctly calculates $\lambda(s,v)$. If $\lambda(s,v) = 1$ then $m_1(v)$ is the incoming arc of the unique maximal $\overline{s}v$ set with $\varrho(X) = 1$. If for the arc uw entering this set X we have $\lambda(s,u) = 2$, then $\{m_2(v), m_3(v)\} = \{m_2(u), m_3(u)\}$. If $\lambda(s,v) = 2$ then $m_2(v), m_3(v)$ is the pair of incoming arcs of the unique maximal $\overline{s}v$ set with $\varrho(X) = 2$.*

Proof. First suppose that $\lambda(s,v) \geq 3$. By Claim 2.6.1, $|M_1(v)| \geq 3$, $|M_2(v)| \geq 3$. and $|M_2'(v)| \geq 3$. Consequently in this case node v correctly concludes $\lambda(s,v) \geq 3$ and it will send $*$s as messages.

Now suppose $\lambda(s,v) = 1$, and let X denote the unique maximal set with $s \notin X$, $v \in X$, $\varrho(X) = 1$, and let uw be the unique arc entering X. In this case clearly $m_1(w) = uw$ (otherwise $m_1(w)$ would be an arc e entering another set Y with $u \in Y$ and $\varrho(Y) = 1$, but then $X \cup Y$ would be a bigger set with one incoming arc). It is easy to see that now along every arc inside X the first message is also uw, so only this message arrives at v as first message

and then v correctly sets $\lambda(s,v) = 1$. Also, if $\lambda(s,u) = 2$, then inductively we may assume that $|\{m_2(u), m_3(u)\}| = 2$ hence $M_2(z)$ remains this set for every node only reachable from u, including v.

Finally suppose $\lambda(s,v) = 2$, and let X denote the unique maximal $\overline{s}v$ set with $\varrho(X) = 2$, and let uw and $u'w'$ be the two arcs entering X. Note that $\lambda(s,u), \lambda(s,u') > 1$, otherwise X would not be maximal. That is, $m_1(u) = m_1(u') = *$. By Claim 2.6.1, $|M_1(v)| \geq 2$, so v does not set $\lambda(s,v)$ to one.

As D is acyclic with a unique source s, and every node is reachable from s, the subgraph of D spanned by X either contains one source, say w, or contains two sources: w and w' (a source must be the head of an entering arc).

Case I $w = w'$. As w is the source of $G[X]$, we have $\varrho(w) = 2$, so $|M_1(w)| = 2$ and $\{m_2(w), m_3(w)\} = \{uw, u'w\}$. Therefore every node x inside X has $M_2(x) = \{uw, u'w\}$. As $|M_2(v)| = 2$, v sets $\lambda(s,v) = 2$.

Case II $w \neq w'$. Let X_1 denote the set of vertices $x \in X$ only reachable from one of w and w'. It follows that $\lambda(s,x) = 1$ for all $x \in X_1$ hence every $x \in X_1$ has $M_1(x) = \{uw\}$ or $\{u'w'\}$. If node v is a source of $G[X \setminus X_1]$, then $M_1(v) = uw, u'w'$. For a node $v \in X \setminus X_1$ with entering arcs from X_1 and also from $X \setminus X_1$, it holds that $\{uw, u'w'\} \subseteq M_2(v)$, since an entering arc a not coming from X_1 is important for v and it carries $\{uw, u'w'\}$ in $\{m_2(a), m_3(a)\}$. An arc b coming from X_1 carries uw or $u'w'$ in $m_1(b)$, so b is not important. Hence $M_2' = \{uw, u'w'\}$. Finally for a node $v \in X \setminus X_1$ with all entering arcs from $X \setminus X_1$, clearly $M_2 = \{uw, u'w'\}$.

□

2.6.3 Experimental results

We compared our heuristic 2-Max for three layers with the heuristic of Kim et al. which they called *minCut* [22].

We generated random acyclic networks with given number of nodes and given arc densities. Then we chose some nodes as receivers with a given probability. Finally for every receiver t we calculated $i = \min(3, \lambda(s,t))$ and put t

randomly into one of the sets $T_1, \ldots, T_i$.

The comparison is not easy, because there is no obvious objective function that measures the quality of the solutions. Generally we can say that none of the algorithms outperformed the other. To illustrate this we show an example, which was run on random networks with 551 nodes and 2204 arcs and with probability 0.1 for selecting receivers. We describe only the number of nodes in T_3 receiving 1,2, or 3 layers (see Figure 2.2).

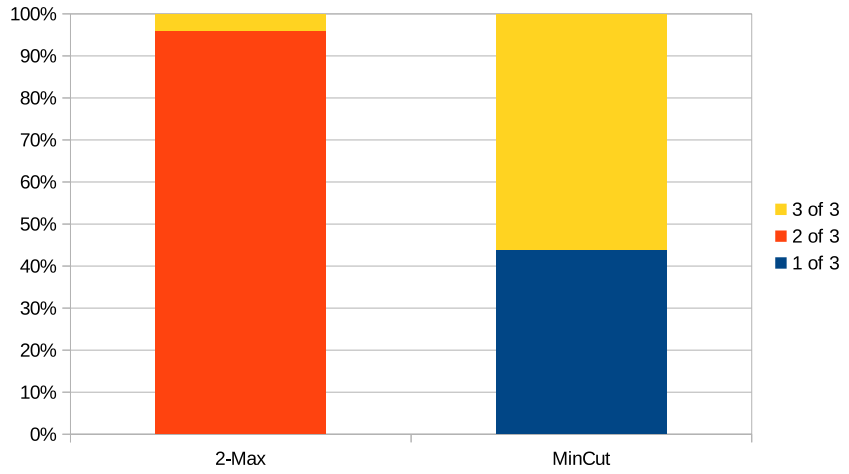

Figure 2.2: Comparison on one specific example for users with demand 3.

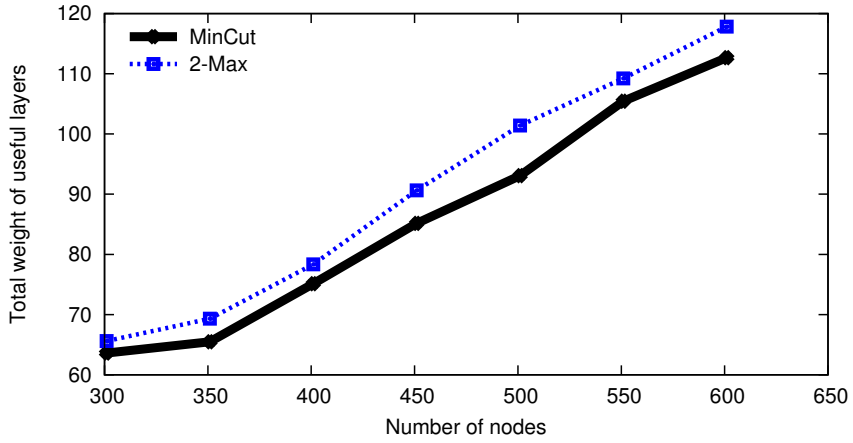

Figure 2.3: Comparison of weighted performances with varying number of nodes.

For making more precise comparison, we had to define a realistic objective function. As both heuristics carry the base layer to every receiver, we did not give a score for these. The objective function we chose is $2 \cdot r_2^2 + 1.8 \cdot r_3^2 + 2.7 \cdot r_3^3$, where r_2^2 is the number of receivers in T_2 that received two layers, r_3^2 is the number of receivers in T_3 that received two layers, and r_3^3 is the number of receivers in T_3 that received three layers. The ideology behind this is the following. A receiver with demand two is absolutely satisfied if it receives two layers. A receiver with demand three is a little bit less satisfied if it receives two layers, but much more happy than one receiving only one layer. And a receiver receiving three layers is 1.5 times more satisfied than one receiving only two.

We made series of random inputs with varying number of nodes. For each node number we generated 10 inputs, calculated the scores defined above, and averaged, this score makes one point in the graphs shown in Figure 2.3. Implementations were carried out with LEMON C++ library [9].

2.7 Performance of randomized height bounding network coding algorithms

The notion of fan-extension was applied in the previous section for feasible network code construction. In this section we show how it can be generalized to determine the expected performance of a family of randomized layered network coding heuristics such as minCut [22], which was already mentioned in the previous section. As an application we give a new proof for the performance guarantee of minCut. Another useful consequence of these results is that the expected performance can be determined without simulations.

2.7.1 Height bounding randomized network coding algorithms

Assume that a multi-layered network coding problem is given as described in Definition 2.2.1. One can get a randomized network coding algorithm from any function $\ell : A \to \mathbb{N}$ the following way.

Definition 2.7.1 (Randomized height bounding heuristic)**.** Assume a fixed finite field $\mathbb{F}_q$, and a function $\ell : A \to \mathbb{N}$ is given, which is called the **height bound function** of the heuristic. Random network codes on the arcs are generated in the topological order of the tails of arcs, depending on the performance $p_{q,\ell}$ of their tails and function ℓ. (Value $p_{q,\ell}(s)$ is defined to be k.)

If network codes are already generated for all entering arcs of a node u, then its performance $p_{q,\ell}(u)$ is calculated. For an arc uv leaving u:

i) if $\ell(uv) \leq p_{q,\ell}(u)$, then $\mathbf{c}(uv)$ is chosen from $\langle \mathbf{e}_1, \ldots, \mathbf{e}_{\ell(uv)} \rangle$ with uniform distribution.

ii) if $\ell(uv) > p_{q,\ell}(u)$, then $\mathbf{c}(uv)$ is a vector $\mathbf{c}_p(uv)$ chosen from $\langle \mathbf{e}_1, \ldots, \mathbf{e}_{p_{q,\ell}(u)} \rangle$ with uniform distribution, plus a random linear combination of global coefficients on all those arcs entering u which have height at least $p_{q,\ell}(u)+1$ and at most $\ell(uv)$. (If there is no such arc and $p_{q,\ell}(u) = 0$, then $\mathbf{c}(uv) = 0$.)

2.7.2 Performance characterization

In this subsection we show that the expected performance and height of a randomized height bounding heuristic converges to a height-performance pair as the field size tends to infinity.

In order to get estimations on $p_{q,\ell}$, we introduce some new notions.

Definition 2.7.2. Let H be a height function in a network. For a set of arcs $R \subseteq A$, let $hd(R)$ (**height-dimension** of R) denote the maximal integer i for which $|\{a \in R |\ j \leq H(a) \leq i\}| \geq i - j + 1$ for every $1 \leq j \leq i$.

Theorem 2.7.1 (B-K, Király)**.** *For a fixed height bound function ℓ and a finite field $\mathbb{F}_q$, let $h_{q,\ell}$ and $p_{q,\ell}$ denote the random height and performance functions of the algorithm, respectively. There exists a height-performance pair (H_ℓ, P_ℓ) on D such that*

- $\lim_{q\to\infty} Prob\big(h_{q,\ell}(uv) = H_\ell(uv)\big) = 1$ *for all* $uv \in A$,
- $\lim_{q\to\infty} Prob\big(p_{q,\ell}(v) = P_\ell(v)\big) = 1$ *for all* $v \in V$.

Moreover, H_ℓ and P_ℓ can be determined in polynomial time.

Proof. We are going to give an inductive proof, calculating H_ℓ and P_ℓ in a fixed topological order. For the source s, $P_\ell(s) := k$. Let us fix a topological order of D, and let v^* denote the last node. Suppose that the inductive assumption holds for the subgraph $D - v^*$. For an arc $uv^* \in A$, value $H_\ell(uv^*)$ can be determined from values earlier in the topological order, as the following lemma shows.

Lemma 2.7.3. *For an arc uv^*,*

i) if $P_\ell(u) \geq \ell(uv^*)$ then $H_\ell(uv^*) := \ell(uv^*)$,

ii) if $P_\ell(u) < \ell(uv^*)$ then $H_\ell(uv^*) := \max\{H_\ell(wu) | wu \in A, H_\ell(wu) \leq \ell(uv^*)\}$).

Proof. Case i): By the inductive assumption, for every $0 < \epsilon < 1$ there exists an N_ϵ such that $\mathrm{Prob}\big(p_{q,\ell}(u) = P_\ell(u)\big) > 1 - \epsilon$ for every $q > N_\epsilon$. When $p_{q,\ell}(u) = P_\ell(u)$, the randomized heuristic over $\mathbb{F}_q$ generates on arc uv^* a global coefficient of height $\ell(uv^*)$ with probability $1 - \frac{1}{q}$, so $\mathrm{Prob}\big(h_{q,\ell}(uv^*) = \ell(uv^*)\big) > (1-\epsilon)(1-\frac{1}{q})$, which proves the claim for this case.

Case ii): Let wu be an arc entering u with $H_\ell(wu) = H_\ell(uv^*)$. By the inductive assumption, for every $0 < \epsilon < 1$ there exists an N_ϵ such that $\mathrm{Prob}\big(h_{q,\ell}(wu) = H_\ell(wu)\big) > 1 - \epsilon$ for every $q > N_\epsilon$. The key observation is that $\mathrm{Prob}\big(h_{q,\ell}(uv^*) = H_\ell(uv^*)\big) > (1-\epsilon)(1-\frac{1}{q})$, which similarly proves this case of the claim. □

The remaining part of the inductive step is the calculation of $P_\ell(v^*)$. Note that we may assume by the inductive assumption that H_ℓ is defined on every arc of A, and P_ℓ is defined on $V - v^*$.

We are going to use an auxiliary digraph $D^* = (V^*, A^*)$ similar, but not identical to the ones defined in Theorem 2.4.3. We are going to see that some random network codes on D^* and D correspond to each other with high probability.

The nodes of D^* are the following: $V^* = V \cup \{t_1, \dots, t_k\} \cup \{z_a^I, z_a^O | a \in A\}$. The arcs of A^* consist of several different sets. First, we add arcs st_i, $1 \leq i \leq k$ to the graph. Then we add $i-1$ parallel $t_{i-1}t_i$ arcs for each $2 \leq i \leq k$, denoted by $(t_{i-1}t_i)^j$ for $1 \leq j \leq i-1$. For each arc $uv \in A$ we add arcs $z_{uv}^I z_{uv}^O$ and $z_{uv}^O v$ to A^*. If an arc $uv \in A$ is free, we add arc $t_{H_\ell(uw)} z_{uw}^I$ to the graph. For a non-free arc $uv \in A$ we add all arcs $z_{xu}^O z_{uv}^I$, where (xu, uv) is a pair in D with $H_\ell(xu) \leq H_\ell(uv)$, and arc $t_{P_\ell(u)} z_{uv}^I$.

Definition 2.7.4. Let a finite field $\mathbb{F}_q$ be fixed. Let c_0 and α_0 denote the following partial global and local coding coefficient functions over $\mathbb{F}_q$, respectively:

$$\begin{cases} c_0(st_i) = \mathbf{e}_i & \text{for } 1 \leq i \leq k \ , \\ c_0((t_i t_{i+1})^j) = \mathbf{e}_j & \text{for } 1 \leq i \leq k \text{ for } 1 \leq j \leq i \ , \end{cases}$$

and

$$\alpha_0(t_i z_{uv}^I, z_{uv}^I z_{uv}^O) = \alpha_0(z_{uv}^I z_{uv}^O, z_{uv}^O z_{vw}^I) = \alpha_0(z_{uv}^I z_{uv}^O, z_{uv}^O v) = 1.$$

A network code $(\alpha^*, \mathbf{c}^*)$ on D^* is an **extension** of $\mathbf{c}_0, \alpha_0$, if $\mathbf{c}^* = \mathbf{c}_0$ and $\alpha^* = \alpha_0$ on the domains of $\mathbf{c}_0$ and α_0, respectively.

We are going to consider **random extensions** of $\mathbf{c}_0, \alpha_0$. Global coefficients on arcs of the form $t_i z_{uv}^I$ are chosen with uniform distribution from $\langle \mathbf{e}_1, \dots, \mathbf{e}_i \rangle$. Local coefficients on arcs of the form $z_{xu}^O z_{uv}^I, z_{uv}^I z_{uv}^O$ are chosen independently with uniform distribution from $\mathbb{F}_q$. These values with $(\alpha_0, \mathbf{c}_0)$ define a random network code on D^*. Let $(\alpha^*, \mathbf{c}^*)$ and (h_{q,c^*}, p_{q,c^*}) denote the resulting random

network code and height-performance pair on D^*.

Let us call a network code $(\alpha, \mathbf{c})$ on D **smooth** if $p_c = P_\ell$ on $V - v^*$ and $h_c = H_\ell$ on A. Similarly, let us call an extension $(\alpha^*, \mathbf{c}^*)$ of $(\alpha_0, \mathbf{c}_0)$ **smooth** if $p_{c^*} = P^*$ on $V - v^*$, where function $P^* : V - v^* \to \{0, 1, \ldots, k\}$ is $P^*(u) = P_\ell(u)$ for $u \in V - v^*$, and $h_{c^*} = H^*$ on A^*, where

$$\begin{cases} H^*(st_i) = i & \text{for } 1 \leq i \leq k\ , \\ H^*((t_{i-1}t_i)^j) = j & \text{for } 2 \leq i \leq k\ 1 \leq j \leq i-1\ , \\ H^*(z^I_{uv}z^O_{uv}) = H^*(z^O_{uv}v) = H_\ell(uv) & \text{for } uv \in A, \\ H^*(t_i z^I_{uv}) = i & \text{for a free arc } uv \in A, \\ H^*(z^O_{xu}z^I_{uv}) = H_\ell(xu) & \text{for a non-free arc } uv \in A. \end{cases}$$

The following lemma shows the relationship between smooth extensions on D^* and smooth network codes on D.

Lemma 2.7.5. *Let a finite field $\mathbb{F}_q$ be fixed. There is a bijection between smooth extensions of $(\alpha_0, \mathbf{c}_0)$ and smooth network codes of D such that $p_{c^*}(u) = p_c(u)$ for every $u \in V$ for every (c^*, c) corresponding network code pair.*

Proof. Similarly to L, let L^* denote the set of pairs in D^*. Let L_H be the subset of those pairs (xu, uv), for which $H_\ell(xu) \leq H_\ell(uv)$ and uv is not free, Let $\phi : A \to A^*$ denote the mapping $\phi(uv) := z^I_{uv}z^O_{uv}$, and let $\psi : L_H \to L^*$ be $\psi(xu, uv) := (z^O_{xu}z^I_{uv}, z^I_{uv}z^O_{uv})$. The lemma is proved by two claims.

Claim 2.7.6. *Let $(\alpha^*, \mathbf{c}^*)$ be a smooth extension of $(\alpha_0, \mathbf{c}_0)$. Then $\mathbf{c}(uv) = \mathbf{c}^*(\phi(uv))$, and $\mathbf{c}_p(uv) = \mathbf{c}^*(t_{P_u}z^I_{uv})$, and $\alpha(xu, uv) = \alpha^*(\psi(xu, uv))$ gives a smooth network code on D.*

Proof. Since for every node $u \in V - v^*$, $p_c(u) = P_\ell(u)$, $\mathbf{c}(uv) \in \langle \mathbf{e}_1, \ldots, \mathbf{e}_{P_\ell(u)} \rangle$ if $H_\ell(uv) \leq P_\ell(u)$. □

We can formulate the inverse of the former claim too.

Claim 2.7.7. *Let $(\alpha, \mathbf{c})$ be a smooth network code on D. Then $\mathbf{c}^*(z^I_{uv}z^O_{uv}) = \mathbf{c}\big(\phi^{-1}(z^I_{uv}z^O_{uv})\big)$, and $\mathbf{c}^*(t_{P_u}z^I_{uv}) = \mathbf{c}_p(uv)$, and $\alpha^*(z^O_{xu}z^I_{uv}, z^I_{uv}z^O_{uv}) = \alpha\big(\psi^{-1}(z^O_{xu}z^I_{uv}, z^I_{uv}z^O_{uv})\big)$ gives a smooth extension of $(\alpha_0, \mathbf{c}_0)$.*

Proof. Since $(\alpha, \mathbf{c})$ is smooth, local coefficients $\alpha(xu, uv) = 0$ for pairs with $H_\ell(xu) > H_\ell(uv)$, so the defined values indeed form a network code on D^*. □

The existence of the bijection follows from Claims 2.7.6 and 2.7.7, proving the lemma. □

By the inductive assumption we know that the probability of a random network code on D being smooth tends to 1. We can formulate an analogue claim for random extensions.

Claim 2.7.8. $\lim_{q\to\infty} \mathit{Prob}\big((\alpha^*, c^*) \text{ is smooth }\big) = 1.$

□

The existence of $P(v^*)$ is proved from the existence of a similar value on D^*.

Lemma 2.7.9. *There exists an integer $P^*(v^*)$ such that*

$$\lim_{q\to\infty} \mathit{Prob}\big(p_{q,c^*}(v^*) = P^*(v^*)\big) = 1.$$

This value can be determined in polynomial time.

Proof. Let $F \subseteq A^*$ denote the set of arcs of type $t_i z^I_{uv}$.

Claim 2.7.10. *For a minimal $\overline{s}v^*$-cut X^* in D^*,*

- $\lim_{q\to\infty} \mathit{Prob}\big(p_{q,c^*}(v^*) \le hd_{H^*}(\Delta^{in}(X^*)\big) = 1.$
- $\lim_{q\to\infty} \mathit{Prob}\big(p_{q,c^*}(v^*) \ge hd_{H^*}(\Delta^{in}(X^*) \cap F\big) = 1.$

Proof. Let B_{v^*} and B_{X^*} denote the subspaces of $\mathbb{F}_q^k$ spanned by the global coefficients on arcs in $\Delta^{in}(v^*)$ and $\Delta^{in}(X^*)$, respectively. Then $B_{v^*} \subseteq B_{X^*}$ always holds. Since $h_{q,c^*} \le H^*$, we get the first part of the claim.

For the second part, since X^* is a minimal cut, there are $\varrho(X^*)$ pairwise arc-disjoint paths from the arcs of $\Delta^{in}(X^*)$ to v^*. Let L_{X^*} denote the set of

pairs of the form $(z^O_{xu}z^I_{uv}, z^I_{uv}z^O_{uv})$ such that $xu, uv \in D^*_{X^*} \cup \Delta^{in}(X^*)$. Local coefficients of pairs in L_{X^*} are are exactly the ones chosen randomly for an extension of $(\alpha_0, \mathbf{c}_0)$ in $D^*_{X^*} \cup \Delta^{in}(X^*)$. Applying Claim 2.7.8 and that all paths in D^* are monotone increasing according to H^*, it can also be proved that $\text{Prob}(B_{v^*} = B_{X^*}) \geq (1 - \frac{1}{q})^{|L_{X^*}|}$, which tends to 1 as q tends to infinity.

Let $f = hd_{H^*}(\Delta^{in}(X^*) \cap F)$. Then $\text{Prob}(\langle \mathbf{e}_1, \ldots, \mathbf{e}_f \rangle \subseteq B_{X^*}) \geq (1 - \frac{1}{q})^f$, which also tends to 1 in this case, giving the proof of the claim.

□

Claim 2.7.11. *There exists a minimal $\overline{s}v^*$-cut X^* in D^* such that*

$$hd_{H^*}(\Delta^{in}(X^*)) = hd_{H^*}(\Delta^{in}(X^*) \cap F).$$

Proof. We show that the minimal $\overline{s}v$-cut in D^* with maximum size defines a cut as described in the lemma. Assume indirectly that $h := hd_{H^*}(\Delta^{in}(X^*)) > hd_{H^*}(\Delta^{in}(X^*) \cap F) := f$. Then there exists an $1 \leq i \leq h$ such that $t_i \notin X^*$. Let V^*_h denote the set of nodes cut from s by node set $\{t_1, \ldots, t_h\}$. From the previous observation $V^*_h \not\subseteq X^*$. Note that all paths in D^* starting from a node t_i are monotone increasing according to H^*, so for every arc uv in $\Delta^{in}(X)$ with $H^*(uv) \leq h$, $u \in V^*_h$ holds. Since $h = hd_{H^*}(\Delta^{in}(X))$, there are at least h such arcs. Then $\Delta^{in}(V^*_h \cup X^*) \leq |\Delta^{in}(X)| - h + h$, so it is also a minimal $\overline{s}v$-cut, contradicting the assumption. This proves the lemma. □

The former claim with Claim 2.7.10 gives that $P^*(v^*)$ exists, and equals $hd_{H^*}(\Delta^{in}(X^*)$. Since X^* can be determined in polynomial time, $P^*(v^*)$ can also be calculated.

□

Now we explain how the existence of $P(v^*)$ follows from Lemma 2.7.9. From the inductive assumption $\lim_{q\to\infty} \text{Prob}\big((\alpha_{q,\ell}, \mathbf{c}_{q,\ell})\ \textit{is smooth}\ \big) = 1$, so combined with Claim 2.7.8 and Lemma 2.7.5 we get that

$$1 = \lim_{q\to\infty} \mathrm{Prob}\big(p_{q,c^*}(v^*) = P^*(v^*)\big) =$$
$$\lim_{q\to\infty} \mathrm{Prob}\big(p_{q,c^*}(v^*) = P^*(v^*) | (\alpha^*, \mathbf{c}^*) \ \textit{smooth}\big) =$$
$$\lim_{q\to\infty} \mathrm{Prob}\big(p_{q,\ell}(v^*) = P^*(v^*) | (\alpha_{q,\ell}, \mathbf{c}_{q,\ell}) \ \textit{smooth}\big) =$$
$$\lim_{q\to\infty} \mathrm{Prob}\big(p_{q,\ell}(v^*) = P^*(v^*)\big)$$

Hence $P(v^*) = P^*(v^*)$, which can be calculated in polynomial time, according to Lemma 2.7.9. This concludes the proof of the theorem. □

2.7.3 Performance guarantees

As an application of Theorem 2.7.1 we give a new proof for the performance guarantee of minCut. First we define the height bounding function $minCut$. Let a multi-layered network coding problem be given. We may assume that all sink nodes are receivers. Values of $minCut$ are calculated in the reverse of a topological order. For entering arcs of a receiver node t with no outgoing arcs, $minCut(wt) := \lambda_D(s,t)$ for all $wt \in A$. If values on all outgoing arcs of a node v are already defined, then $minCut(uv) := \min\{minCut(vw) | vw \in A\}$ if this minimum is greater than $\lambda_D(s,v)$, and $\lambda_D(s,v)$ otherwise. In the latter case let us put node v to set U.

Theorem 2.7.2 (Kim, Lucani, Shi, Zhao, Médard [22]). *If the fieldsize is large enough, minCut sends the first layer to every receiver with high probability.*

In fact, we prove a slightly more general statement.

Theorem 2.7.3 (B-K, Király). $P_{minCut}(t) \geq 1$ *for all* $t \in T \cup U$ *and* $H_{minCut}(uv) \geq 1$ *for all* $uv \in A$.

Proof. We prove the theorem inductively, in the topological order of the nodes. Suppose that the statement holds until node v^*. Applying Lemma 2.7.3 to an entering arc uv^* and the inductive assumption that $H_{minCut}(wu) \geq 1$ for every $wu \in A$, we get that $H_{minCut}(uv^*) \geq 1$.

Assume that $v^* \in T \cup U$.

Claim 2.7.12. $P_{minCut}(v^*) \geq 1$.

Proof. We will use the notations of Claim 2.7.11. From the lemma we know that there exists a minimal $\overline{s}v^*$-cut X^* in D^* with $|X^*|$ maximal. Assume indirectly that $hd(\Delta^{in}(X^*) \cap F) = 0$. Then there are no st_i arcs in $\Delta^{in}(X^*)$. If $\lambda_{D^*}(s, v^*) = \lambda_D(s, v^*)$, then we get that $P_{minCut}(v^*) = \lambda_D(s, v^*) \geq 1$. If $\lambda_{D^*}(s, v^*) < \lambda_D(s, v^*)$, since there are fewer sv^* paths in D^* than in D, there is a pair of consecutive arcs (wu, ux) with $H_{minCut}(wu) > H_{minCut}(ux)$ such that $z^I_{ux} \in X^*$ but $z^O_{wu} \notin X^*$. Then $u \in T \cup U$, so from the inductive assumption $P_{minCut}(u) \geq 1$. But then arc $t_1 z^I_{ux} \in \Delta^{in}(X^*)$, contradicting the assumption that $hd(\Delta^{in}(X^*) \cap F) = 0$. □

The claim concludes the proof of the theorem. □

2.7.4 Open problems

We mention two possible topics for future research. Let $\mathcal{F} \subseteq 2^k$ denote the set of allowed requests a receiver may have. In the multi-layered network coding problem $\mathcal{F} = \{[0], [0, 1], [0, 1, 2], \ldots, [0, 1, \ldots, k]\}$. It is a possible generalization to consider demands with a laminar structure, when for each pair of sets $X, Y \in \mathcal{F}$ either $X \subseteq Y$ or $Y \subseteq X$.

Another possible direction is to modify the height bounding randomized heuristic framework. Let B_v denote the subspace of $\mathbb{F}_q$ spanned by the global coefficients on arcs entering node v. Then we may choose $\mathbf{c}(vw)$ randomly from $B_v \cap \langle \mathbf{e}_1, \ldots, \mathbf{e}_{\ell(vw)} \rangle$.

Chapter 3

Wireless multi-layer multicast

This chapter summarizes work carried out with Pedersen, Lucani and Fitzek during a visit at Aalborg University [28]. The main focus was to explore possible applications of multi-layer network coding, presented in the previous chapter, in a real-world setting. Results presented are more practical than theoretical, compared to other chapters.

3.1 Introduction

Efficient video delivery for devices with heterogeneous requirements and capabilities has posed significant challenges from a network use perspective. Although it is possible to deliver different video qualities to different users by using separate data streams, this solution is highly inefficient as it does not exploit the inherent dependencies of these data streams. Multiple Description Coding (MDC) and Scalable Video Coding (SVC) have provided alternatives to cater to users with different quality demands.

More recently, network coding has shown an interesting potential for enhancing the performance of layered schemes for achieving higher throughput in the network, e.g., [22, 24, 44] or compensating for inherent packet losses in wireless environments, e.g., [29]. In particular, work in [22] studied the case of layered multicast on wireline networks proposing a simple message passing algorithm to solve the demands of multiple receivers and exploiting on demand decoding at intermediate nodes for enhanced performance. [24] provided a generalization to the approach in [22] presenting an algorithm

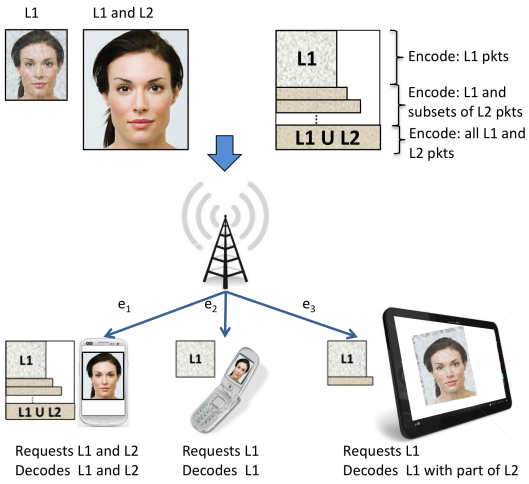

Figure 3.1: Basic broadcast topology and encoding decoding matrices

that solves the problem for two layers optimally for certain natural objective functions as well as useful heuristics for the case of three layers. The work in [44] provided heuristics for coding across multiple layers, as [22], but without allowing for decoding at interior nodes and assuming full knowledge of the network's topology. [43] studied the joint design of multi-resolution codes and network coding while [29] provided network coding structures for better delay/reliability in the presence of multi-layer codes for video applications. Random linear coding strategies with overlapping and non-overlapping time windows are compared in the multi-layered setting in [42]. Optimization of rateless code schemes for diverse users were studied in [17] and [33].

In a wireless multi-layered multicast setting, receivers with different computational power and demands make use of different types of encoded packets. We present a scheme that splits higher layers into sublayers and sends inter and intra-layer packets with different probabilities. The advantage of this flexibility is that it can increase the coding advantage of users with low-demand by extracting information from inter-layer packets. To the best of our knowledge, the effect of layer sizes on this coding advantage has not been investigated. The scheme takes the synthesis of users into account and determines its parameters, sublayer sizes and probabilities, based on user preferences.

The rest of this chapter is organized as follows. Section 3.2 describes the wireless multicast problem, and Section 3.3 presents the proposed solution with formulas for the performance of different user types. Performance comparison is discussed in Section 3.4 and numerical results are presented in Section 3.5.

3.2 Problem Formulation and Contributions

Network coding requires an increased computational complexity from user devices, which may have limitations on its applicability. Increasing the number of packets encoded together also increases the complexity of the decoding phase. Network coding algorithms for multi-layer content typically distinguish two types of coded packets, according to the number of layers the packet contains data from:

- **Intra-layer** packets contain data from one single layer only, in our case the base layer.
- **Inter-layer** packets may contain encoded packets from several layers, and they require higher computational capacity.

Users may have different preferences on the type of the encoded packets. Figure 3.1 shows an example with two layers and presents three user types with different demands and computation abilities. User 1 requests two layers because of its screen with high resolution and its computational capability to decode inter-layer packets. Thus, it exploits inter-layer packets mixing the two layers for recovering both available layers. User 2 has a lower computation power and only requests the first layer, so it only exploits intra-layer packets containing the base layer only. Finally, User 3 also requests the first layer only due to its screen limitation. However, since it is willing to invest additional computational effort to get a better service, it will also extract information from inter-layer packets. In our example, User 3 only exploits a part of the inter-layer packets.

We introduce a scheme for wireless multi-layer multicast which takes heterogeneity of users into account. It addresses the problem of finding the trade-off between sending intra-layer packets of the base layer, and inter-layer packets mixing the base layer and one refinement layer. In addition to the concept of mixing inter- and intra-layer packets, we divide higher layers into sublayers. An inter-layer packet contains encoded packets from the base layer and some encoded packets from one specific sublayer. The reason for the concept of sublayers is that it decreases computational complexity and increases useful information extracted from inter-layer packets for User 3.The nature of the analysis and implementation using sub-layers allows us to easily map scenarios with more than two layers into our overall solution.

The overall goals of our work are the following:

- Reduce (and make more deterministic) the time to get the base layer for all receivers as well as reducing their time to recover all desired layers.
- Exploit the inherent, heterogeneous computing capabilities of different devices to improve their overall performance.
- Provide a single encoding structure that allows heterogeneous receivers to improve their service quality. Since we assume the different data packets have the potential to be received at each destination, these destinations should have the ability to use them if needed.
- Provide an explicit trade–off in performance between different types of receivers.

3.3 Proposed Scheme

We consider a source S transmitting coded packets. The data is split in n layers, namely, Layer 1 (L_1), Layer 2 (L_2), ..., Layer n (L_n) where l_i packets compose layer i. We say that L_i is higher than L_j if $i > j$. Correspondingly, if $i < j$, then Layer i is lower. In order to use L_i, a receiver needs to also decode

all the data packets corresponding to lower layers. The source creates linear combinations of only L_1 packets with probability p_1, while with probability p_i it will generate coded packets involving all L_1 packets and some or all of the L_i packets. The latter contains several cases, where we divide L_i in K_i sublayers of size d_i packets, each sublayer with probability $1/K_i$ is to be chosen. The reason for this code structure is that there are different L_1 receivers. For example, L_1 receivers with limited computing capabilities will only use L_1 packets. However, L_1 receivers with more computational resources, e.g., a mobile device with a fast processor but a small screen, can exploit some of the combinations of L_1 and L_i packets. Our goal is in part to characterize the appropriate p_i and K_i to improve performance of the different receiver types. Note that a larger K_i will benefit L_1 receivers with additional computing capabilities, because they will be able to decode Layer 1 without getting all degrees of freedom to decode both L_1 and L_i. However, a larger K_i makes for a less efficient code, i.e., requiring more coded packets to decode both L_1 and L_i.

The choice of $1/K_i$ as the probability to choose sub-layer i is optimal for cases where all receivers have the same channel loss probability. This choice can be modified in the event of channel asymmetries or if some sub-layers are known to be discarded by all devices interested in L_1. However, this optimization is out of the scope of our current work.

3.3.1 Preliminaries

For our analysis, we make the following assumptions

- **Large Finite Fields:** Arithmetic operations are performed in a finite field with a large number of elements. Thus, a coded packet of a specific sub-layer will provide an independent linear combination if the rank at the receiver can be increased with any coded packet of the given sub-layer.

- **Minimal Feedback from Receivers:** Receivers provide only mini-

malistic feedback indicating that the receiver has successfully decoded its intended layer(s). This allows the system to manage a large number of receivers with limited signaling.

- **Communication Channel:** Transmissions are broadcasted to different nodes in the network. Unless stated otherwise, we focus on the case of a wireless, single hop broadcast network as in Figure 3.1. Packet losses are assumed to be independent.

3.3.2 Encoding, Recoding and Decoding Approaches

The following descriptions are based on our implementation of the algorithms in the Kodo [38] network coding library. The implementation and simulations used in this chapter can be downloaded as a standalone package from [16].

- **Encoder:** In order to implement the layered encoding we used a simple scheme requiring only three minimal changes to an existing RLNC encoder. 1) Before encoding a symbol randomly select a coding layer L_m according to the layer probabilities p_m, where $0 \leq m \leq n$. 2) Generate only non-zero coding coefficients up until the size d_m of the chosen coding layer. 3) Include the layer index into the encoded symbol allowing the decoder to easily identify which layer was used for the encoding.

- **Decoder:** In order to implement the proposed scheme we needed to construct a decoder capable of decoding a specific layer L_i while utilizing j out of a total n layers, where $i \leq j \leq n$. As with the encoder this goal was achieved in three stages (see Fig. 3.2). 1) Extract the layer index of the incoming symbol. If the layer index is larger than j discard the symbol. 2) Otherwise pass the symbol to the *elimination decoder*. The purpose of the elimination decoder is to remove the L_j contribution in the incoming symbols so that it becomes useful for decoding layer L_i. 3) If the elimination decoder successfully removed the L_j contribution

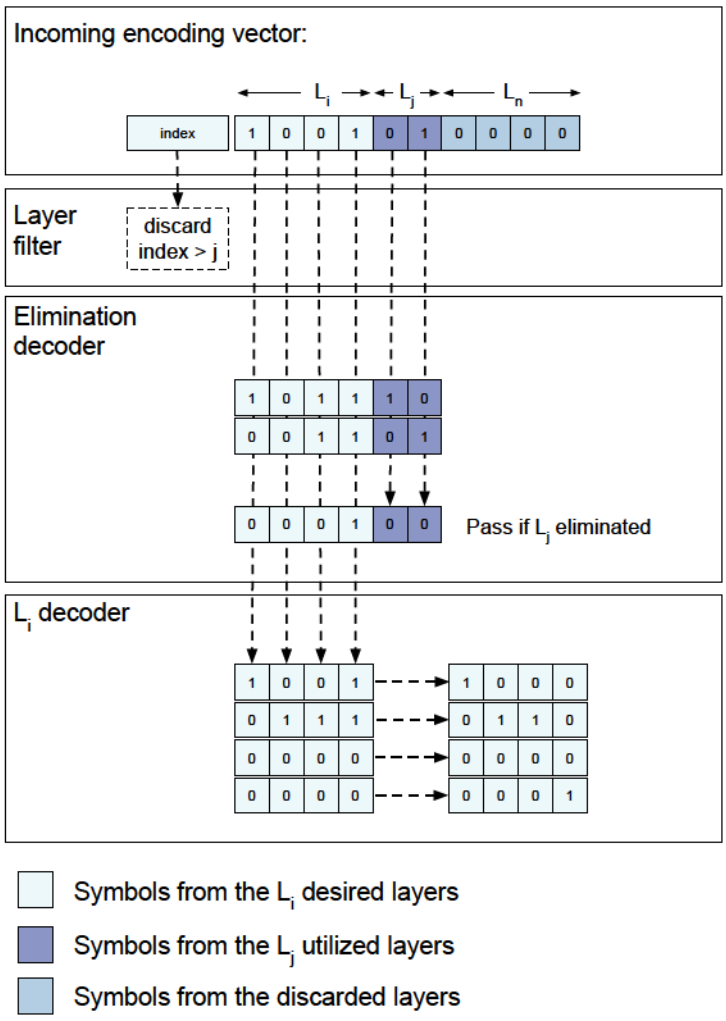

Figure 3.2: Decoding procedure for an L_i decoder with $d_i = 4$, $d_j = 6$ and $d_n = 10$. Symbols are passed through three stages where the first two only conditionally forward the symbol. Decoding is complete when the L_i decoder reaches full rank.

from the incoming symbol it can be passed to the L_i decoder for actual decoding. With this structure we are able to deal with all choices of L_i, L_j and L_n.

- **Recoder:** Recoding at intermediate nodes without altering the coding structure requires the system to control which sub-layers can be combined for generating a coded packet of a given sub-layer. A sim-

ple approach lies in creating a random linear combinations of all coded packets of that sub-layer and sub-layers that have less data packets. This exploits the structure of the source's stream to preserve such structure. Clearly, this recoding procedure benefits higher sub-layers. A more advanced and computationally demanding approach is to perform partial decoding of higher (sub-)layers in order to exploit packets from these in the recoding of lower (sub-)layers.

Remark 3.3.1. Although we study the case of two layers, the management of multiple layers is straightforward in terms of the encoding, recoding, and decoding schemes. The reason is that we are inherently defining sub-layers for layer L_2. Some of these sub-layers can also be full layers in future settings. Clearly, changes in the probabilities of sending each sub-layer will change to provide the desired service.

3.3.3 Delay Performance of Different Receiver Types

In this subsection, we give exact values for the expected number of packages users need to receive in order to be able to decode the demanded layer(s). We present formulas for all the three types of users presented in Section 3.2. Calculations can be extended for the general case of n layers applying similar techniques.

Definition 3.3.2. A packet is called a **1-packet, 2-packet** and **1-2-packet** if it is an encoded packet from original packets of L_1, L_2, and both layers, respectively.

Let x_1 denote the total number of coded packets received when L_1 becomes decodable for a receiver using only 1-packets. Then, since the last packet received must be a 1-packet and the number of previously received packets has a binomial distribution,

$$\Pr(x_1 = n) = \binom{n-1}{l-1} p^{l_1} (1-p)^{n-l_1}$$

The expected value of x_1 can be expressed with the following formula:

$$E(x_1) = \sum_{n=l_1}^{\infty} n \binom{n-1}{l-1} p^{l_1} (1-p)^{n-l_1}$$

Definition 3.3.3. For a 1-2-packet m let m_1 and m_2 denote the 1-packet and 2-packet reduced from m by taking the coefficient vectors of only L_1 and L_2, respectively. Similarly, we define M_1 and M_2 for a set M of 1-2-packets. For such a set M, let $\mathrm{SP}(M) := (|M| - rank(M_2))^+$ be called the **surplus** of M.

Let $\Pr_{SP}(N, K, d, b)$ denote the probability that a random set of 1-2-packets has surplus exactly b. Note that the surplus of such a set is the sum of surpluses of K disjoint subsets containing packets from a certain division. Then, $\Pr_{SP}(N, K, d, b)$ can be calculated recursively for $K > 1, N > 0$:

$$\Pr_{SP}(N, K, d, b) = \sum_{n=0}^{d+b} \binom{N}{n} \frac{1}{K^n} \left(1 - \frac{1}{K}\right)^{N-n} \Pr_{SP}(N-n, K-1, d, b'),$$

where $b' = b - (n-d)^+$.
Similarly, for $b > 0$ let $\Pr^*_{SP}(N, K, d, b)$ denote the probability that a set of N random 1-2-packets has surplus exactly b and the last packet m increases the surplus, that is $\mathrm{SP}(M) > \mathrm{SP}(M - m)$. Then, we have

$$\Pr^*_{SP}(N, K, d, b) = \sum_{n=d+1}^{d+b} \binom{N-1}{n-1} \frac{(K-1)^{N-n}}{K^{N-1}} \Pr_{SP}(N-n, K-1, d, b-(n-d)).$$

Now we are ready to express the expected number of packets a receiver needs for decoding, if both 1-packets and 1-2-packets are used. Let $x_{1|2}$ denote the number if packets received when L_1 becomes decodable. Note that $x_{1|2} \leq l_1 + l_2$. Then, according to whether the last packet is a 1-packet or a 1-2-packet we can distinguish between two cases.

$$\Pr(x_{1|2} = N, \text{last is 1-p.}) = \sum_{n=1}^{l_1} \binom{N-1}{n-1} p^n (1-p)^{N-l_1} \Pr_{SP}(N-n, K, d, l_1-n)$$

$$\Pr(x_{1|2} = N, \text{last is a 1-2-p.}) = \sum_{n=0}^{l_1} \binom{N-1}{n} p^{l_1}(1-p)^{N-l_1} \Pr^*_{SP}(N-n, K, d, l_1-n)$$

$$E(x_{1|2}) = \sum_{N=l_1}^{l_1+l_2} N\Big(\Pr(x_{1|2} = N, \text{last is 1-p.}) + \Pr(x_{1|2} = N, \text{last is 1-2-p.})\Big)$$

Let $\Pr_{SP2}(N, K, d, b)$ denote the probability that a set of N random 1-2-packets has surplus at least b and all the divisions are decodable.

$$\Pr_{SP2}(N, K, d, b) = \sum_{n=d}^{N-(K-1)d} \binom{N}{n} \frac{(K-1)^{N-n}}{K^N} \Pr_{SP2}(N-n, K-1, d, (b-(n-d))^+)$$

Let $\Pr^*_{SP2}(N, K, d, b)$ denote the probability that a random set of N 1-2-packets has surplus at least b, all the divisions are decodable and the last message completes a division.

$$\Pr^*_{SP2}(N, K, d, b) = K\binom{N-1}{d-1}\frac{1}{K^d}\left(1-\frac{1}{K}\right)^{N-d} \Pr_{SP2}(N-d, K-1, d, b)$$

Let $\Pr_{exSP2}(N, K, d, b) =$ denote the probability that a random set of N 1-2-packets has surplus exactly b and all divisions decodable.

$$\Pr_{exSP2}(N, K, d, b) = \sum_{n=d}^{d+b} \binom{N}{n} \frac{(K-1)^{N-n}}{K^N} \Pr_{exSP2}(N-n, K-1, d, b-(n-d))$$

Let x_{12} denote the number of packets needed to decode both layers. According to the type of the last packet there are three cases:

i, last packet is a 1-packet

ii, last packet is a 1-2-packet and L_1 is completed with this packet

iii, last packet is a 1-2-packet and a division in L_2 is completed with this packet

Note that cases (1) and (2) imply that $x_{12} = l_1 + l_2$. Hence, $x_{12} > l_1 + l_2$ implies case (3).

$$\Pr(x_{12} = l_1 + l_2 \text{ and case i,}) = \sum_{n=1}^{l_1} \binom{l_1 + l_2 - 1}{n-1} p^n (1-p)^{l_1+l_2-n} \Pr_{exSP2}(l_1 + l_2 - n, K, d, l_1 - n)$$

$$\Pr(x_{12} = l_1 + l_2 \text{ and not case i,}) = \sum_{n=0}^{l_1} \binom{l_1 + l_2 - 1}{n} p^n (1-p)^{l_1+l_2-n} \Pr_{exSP2}(l_1 + l_2 - n, K, d, l_1 - n)$$

$$\Pr(x_{12} = N > l_1 + l_2) = \sum_{n=0}^{N-l_2} \binom{N-1}{n} p^n (1-p)^{N-n} \Pr{}^*_{SP2}\left(N, K, d, (l_1 - n)^+\right)$$

$$E(x_{12}) = (l_1 + l_2)\Pr(x_{12} = l_1 + l_2) + \sum_{N=l_1+l_2+1}^{\infty} N \Pr(x_{12} = N > l_1 + l_2)$$

3.3.4 Optimization Criteria

The system's optimization criteria can depend on the requirements of the wireless system. For example, if the goal is to minimize the time of the reception of a video frame (with different available layers), the goal is to make all receivers decode at the same time their respective data. On the other hand, if the goal is to optimize energy consumption of the system, then the use of different sub-layers for decoding will affect the computational effort (and processing energy) of the individual users and of the system as a whole.

The key of our approach is that not only p_1 can be used as the variable to tune performance, but rather one of a large group including the p_i choices

for the different sub-layers as well as the number and size of each sub-layer.

Our goal is not to provide a comprehensive discussion of the different optimization options, but rather to show that our procedure opens the door to more flexible and practical optimizations.

3.4 Performance Comparison

In this section, we simulate the performance of the three different user types introduced in Section 3.2 and compare it to the analytical results obtained in the previous sections.

The basic setup of the simulator is shown in Fig. 3.3. As shown a single source is broadcasting to the three users, each packet sent is lost with independent loss probability e_1, e_2 and e_3. One receiver uses only L_1 coded packets, one uses k_{use} sub-layers to help in decoding L_1, and the latter gathers all coded packets to recover L_1 and L_2. Although in our analysis we focused on the case with $k_{use} = K$, we shall explore in more details these options for receivers interested in L_1.

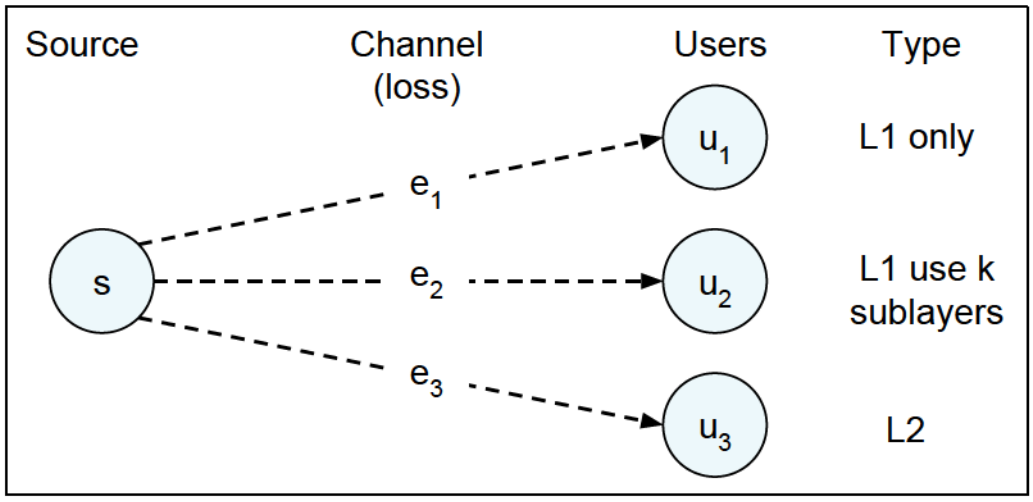

Figure 3.3: Simulator setup with a single source s broadcasting to three users u_1, u_2 and u_3 with three different decoding requirements. During transmission packets are lost with independent erasure probabilities e_1, e_2 and e_3.

In our numerical results, we use as key performance metrics the mean number of received packets by each receiver type in order to decode its intended layer(s) and also the mean total number of transmissions to satisfy all three receivers.

3.5 Numerical Results

This section provides numerical results using the implementation described in Section 3.4 and the analysis from Section 3.3.3 to both confirm our analytical results and illustrate the potential of the proposed mechanism.

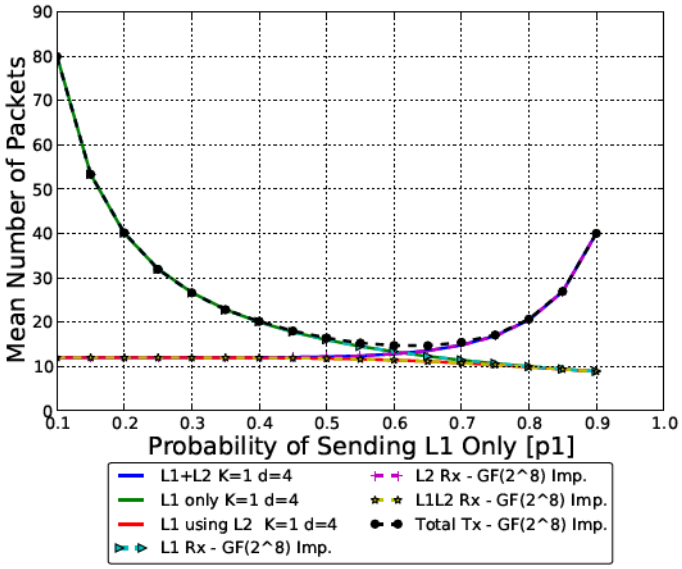

Figure 3.4: Comparing theory and implementation for the number of received packets per receiver when L_2 has 4 packets and L_1 has 8 packets and $K = 1$.

Figure 3.4 shows the performance of the three type of receivers when $K = 1$ and compares the analysis results with the implementation with [16] when using $GF(2^8)$ for its finite field operations. On the one hand, this figure shows that the theoretical and practical results match. On the other hand, it shows that receivers with high computational power can recover L_1 significantly faster than receivers using only L_1 for all values of p_1. If the system attempts

to minimize the overall completion time, a $p_1 \approx 0.6$ will be chosen to strike a balance between L_1 and L_2 receivers. However, an L_1 receiver to decode 30% faster if it exploits inter-layer packets for the same setting.

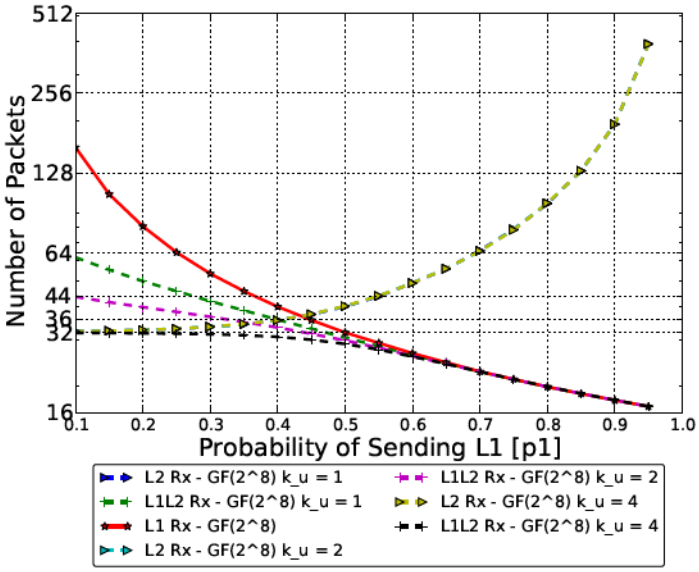

Figure 3.5: Number of received packets before decoding from the three receiver types, when each layer has 16 packets, and dividing L_2 into four sub-layers each of size four packets. k_{use} indicates how many of the sub-layers are being used by the receiver interested in L_1 but exploiting part of L_2.

Figure 3.5 shows that splitting into $K = 4$ sub-layers but letting the receivers decide how many of them (k_{use}) to use to decode L_1 allows for reducing the number of received coded packets before decoding. As shown in the figure, even the use of a single sub-layer, i.e., $k_{use} = 1$, improves significantly the performance of receivers attempting to recover L_1 without requiring a large increase in the processing complexity. In this case, these receivers would need to decode 20 data packets instead of 16 data packets.

More importantly, Figures 3.4 and 3.5 show that our novel encoding structure allows for the system to have more predictable and controllable behavior for L_1 receivers. This also means that the system is less sensitive to the choice of p_1 to determine overall performance. Furthermore, our structure provides more degrees of freedom for the receivers and the transmitter to optimize

the overall system performance while exploiting each device's heterogeneous capabilities. In this way, more powerful receivers with adverse channel conditions can exploit additional processing for coping with their current channel and attain a better service quality. A key aspect is that the receiver could make this choice independently from other devices' policies.

3.6 Perspectives

Although we analyzed the case of no feedback or minimalistic feedback without altering our policy, it is worthwhile to study more dynamic policies. Namely, the probability p_1 (or of any sublayer of Layer 2) could be changed after a group of receivers has finished. For example, p_1 could be made zero if all computationally limited receivers have been satisfied, thus allowing for a more efficient code structure for the remaining receivers. Clearly, this requires that the system knows about which users are actively receiving the data.

Chapter 4

Fixed local coefficients

In this chapter, we investigate various linear network coding problems with partially predetermined coding coefficients. The first version of this problem, called deterministic network coding, was introduced by Harvey, Karger and Murota [19]. They reduced both the unicast and multicast cases to matrix completion. In order to avoid confusion of 'deterministic algorithm' and 'deterministic network coding problem' we call the latter the network code completion problem (NCCP) throughout the chapter.

We also define a related new problem, called 'fixable pairs problem'. We give a sufficient condition for a subset of coding coefficients that can be fixed *arbitrarily* to nonzero values, such that the remaining coefficients can be chosen properly to attain a feasible network code. We present applications of this model, and give necessary and sufficient conditions for the solvability of network coding problems in heterogeneous networks.

In Section 4.1 we present deterministic and randomized algorithms for the NCCP and present applications in wireless relay network problems. In Section 4.2 we discuss the problem of fixable pairs and present some applications on heterogeneous networks.

4.1 Network Code Completion Problem

Suppose that a network coding problem is given.

Definition 4.1.1. For a subset of pairs $M \subseteq L$, a mapping $\alpha_0 : M \to \mathbb{F}_q$ is **extendable**, if there exist local coefficients α of a feasible network code

such that $\alpha = \alpha_0$ on M. Given a network coding problem with a subset of pairs $M \subseteq L$ with a mapping $\alpha_0 : M \to \mathbb{F}_q$, the **network code completion problem** is to decide whether α_0 is extendable.

4.1.1 Deterministic Algorithm

The multicast NCCP is equivalent to determining the simultaneous max rank completion of the transfer matrices, and if the field size is greater than the number of matrices given, then the matrices have a simultaneous max rank completion as proved in [19, 23]. This result can be reformulated as follows.

Theorem 4.1.2 (Harvey, Karger, Murota [19])**.** *If $q > |T|$, a mapping is extendable over $\mathbb{F}_q$ if and only if for every $t \in T$ it is extendable for the one-element terminal set $\{t\}$. Such an extension can be found in polynomial time.*

We give another, simple proof for this theorem. We use the polynomial time algorithm of [19] for the unicast case as a subroutine.

Proof. For a terminal $t \in T$, let α_t denote the extension of α_0 which is feasible for t and let $\mathbf{c}_t : A \to \mathbb{F}_q^k$ denote the corresponding global coefficients. We start by defining $\alpha(\ell) = \alpha_0(\ell)$ for each $\ell \in M$. Let $\ell_1, \ldots, \ell_p$ be an arbitrary order of the pairs in $L \setminus M$. We will determine a value $\alpha(\ell_i)$ for each ℓ_i in this order maintaining that the following mappings α_t^i are feasible for every t.

$$\alpha_t^i(\ell) = \begin{cases} \alpha(\ell) & \text{if } \ell \in M \cup \{\ell_1, \ldots, \ell_i\}, \\ \alpha_t(\ell) & \text{otherwise.} \end{cases}$$

To show the existence of an appropriate $\alpha(\ell_i)$ we prove some lemmas.

Lemma 4.1.3 (B-K, Király [25])**.** *Let $\alpha, \mathbf{c}$ denote the local and global coding coefficients of a network code, respectively. By altering a local coefficient $\alpha(uv, vw)$ to $\alpha'(uv, vw) = \alpha(uv, vw) + \beta$, the new global coefficients have the*

form $\mathbf{c}'(a) = \mathbf{c}(a) + \delta_a \beta \mathbf{c}(uv)$ *with an appropriate value* $\delta_a \in \mathbb{F}_q$ *on every arc* $a \in A$.

Proof. We prove by induction on the topological order of the tails of the arcs. If the tail of an arc is earlier in the order than v, then $\mathbf{c}'$ remains $\mathbf{c}$ and the claim clearly holds. For arcs leaving v, the only arc where $\mathbf{c}$ changes is arc vw, and the claim is again clear. Suppose that the claim holds for every arc with tail before $z \in V$ and let $zx \in A$ be an arc. From the linear combination property we have $\mathbf{c}'(zx) = \sum_{yz \in A} \alpha(yz, zx)\mathbf{c}'(yz) = \sum_{yz \in A} \alpha(yz, zx)(\mathbf{c}(yz) + \delta_{yz}\beta\mathbf{c}(uv)) = \mathbf{c}(zx) + \beta\mathbf{c}(uv) \sum_{yz \in A} \alpha(yz, zx)\delta_{yz}$, which proves this lemma. □

Lemma 4.1.4 (B-K, Király [25])**.** *Let vectors* $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{F}_q^k$ *form a basis of* $\mathbb{F}_q^k$ *and let* $\mathbf{v} \in \mathbb{F}_q^k$, $\delta_1, \ldots, \delta_k \in \mathbb{F}_q$. *Then there is at most one value* $\beta \in \mathbb{F}_q$ *satisfying that* $\{\mathbf{v}'_i = \mathbf{v}_i + \delta_i\beta\mathbf{v}\}_{i=1}^k$ *is not a basis.*

Proof. If every δ_i is zero then the statement is obvious, so without loss of generality we can assume that $\delta_1 \neq 0$. By subtracting $(\delta_i/\delta_1)\mathbf{v}'_1$ from each $\mathbf{v}_i$ $(i = 2, \ldots, k)$, we get that vectors $\{\mathbf{v}'_i\}_{i=1}^k$ form a basis if and only if $\{\mathbf{v}_1 + \delta_1\beta\mathbf{v}\} \cup \{\mathbf{v}_i - (\delta_i/\delta_1)\mathbf{v}_1\}_{i=2}^k$ does. Since $\{\mathbf{v}_1\} \cup \{\mathbf{v}_i - (\delta_i/\delta_1)\mathbf{v}_1\}_{i=2}^k$ is a basis, the lemma follows from Claim 1.2.7. □

Let $\mathbf{c}_t^i$ denote global coefficients corresponding to α_t^i.

Lemma 4.1.5 (B-K, Király [25])**.** *Suppose that the values* $\alpha(\ell_1), \ldots, \alpha(\ell_{i-1})$ *are chosen such that* $\mathbf{c}_t^{i-1}$ *is feasible for* t. *Then for each* t, *there is at most one choice of* $\alpha(\ell_i)$ *such that* $\mathbf{c}_t^i$ *is not feasible for* t.

Proof. Since $\mathbf{c}_t^{i-1}$ is feasible, there is a k-element arc set $B_t = \{b_1, \ldots, b_k\}$ entering t on which the global coefficients of $\mathbf{c}_t^{i-1}$ form a basis. Let $\ell_i = (uv, vw)$. Mappings α_t^{i-1} and α_t^i differ in at most one value (on ℓ_i), hence from Lemma 4.1.3 we get that global coefficients on arcs in B_t have the following form: $\mathbf{c}_t^i(b_j) = \mathbf{c}_t^{i-1}(b_j) + \delta_{b_j}(\alpha_t^i(l_i) - \alpha_t^{i-1}(l_i))\mathbf{c}_t^{i-1}(uv)$. Lemma 4.1.4 says that there is at most one value of ℓ_i such that these vectors do not form a basis. □

Since the size of the field is greater than $|T|$, indeed, a good value can be chosen for each $\alpha(\ell_i)$. If $i = p$ then for every t we have $\alpha_t^p = \alpha$, and we maintained the feasibility. This completes the proof of Theorem 4.1.2. □

Remark 4.1.6. From Lemma 4.1.5 we also get that given a feasible network code and a subset $X \subseteq \mathbb{F}_q$ such that $|X| > |T|$, every local coefficient can be changed to a value from X such that the resulting network code remains feasible.

Remark 4.1.7. In contrast to most previous approaches, our proof can be applied on any acyclic network and hence is more general than former algorithms for deterministic relay networks, which work on layered acyclic graphs only.

Remark 4.1.8. It is open whether this idea can be adapted for the cyclic case. Such an approach may give a faster algorithm than the one given by Erez and Feder in [15] for the multicast case.

4.1.2 Randomized Algorithm

We use the proof presented in the previous subsection to give a new randomized algorithm for the problem which can be applied over every finite field $\mathbb{F}_q$ with $q > |T|$. Let a multicast NCCP be given. Applying results in [20], Kim and Médard [23] gave a lower bound on the probability of a random network code to be feasible over $\mathbb{F}_q$ in the model of [3].

Lemma 4.1.9 (Kim, Médard, [23])**.** *If $q > |T|$ and a mapping $\alpha_0 : M \to \mathbb{F}_q$ has a feasible extension, then the probability that a random extension is feasible, is at least $\left(1 - \frac{|T|}{q}\right)^{|A'|} \geq 1 - \frac{|T| \cdot |A'|}{q}$, where A' is the subset of arcs which appear as a second arc in a pair in $L \setminus M$.*

The idea of our Las Vegas algorithm is to first construct a random extension over a bigger field $\mathbb{F}_{q^r}$ of size q^r such that $\mathbb{F}_q$ is a subfield of $\mathbb{F}_{q^r}$. From Lemma 4.1.5 we can deterministically modify it to get another extension over $\mathbb{F}_q$ if $q > |T|$. Let α be a random extension of α_0 over $\mathbb{F}_{q^r}$. If we choose r such

that $q^r > 2|T| \cdot |A'|$, then from Lemma 4.1.9, α is feasible with probability at least one half. If the extension is not feasible, we repeat generating other random extensions till success. The expectation of the number of extension generations is two. If α is feasible, the extension over $\mathbb{F}_q$ is constructed by choosing for all $\ell \in L \setminus M$ one-by-one a value f from $\mathbb{F}_q$ such that α, changed only on ℓ to $\alpha(\ell) = f$, remains a feasible network code.

Theorem 4.1.10 (B-K, Király [25])**.** *If $q > |T|$, and there exists a feasible extension over $\mathbb{F}_q$ then the algorithm finds one with probability one in polynomial expected running time.*

Proof. We only need to show that a suitable value from $\mathbb{F}_q$ can be chosen for every pair $\ell \in L \setminus M$. In Lemma 4.1.3 we gave a formula, how the modification of a local coefficient influences global coefficients. Combined with Lemma 4.1.4, for each $t \in T$ we get that there is at most one value $f \in \mathbb{F}_{q^r}$ such that changing the value of $\alpha(\ell)$ to f destroys feasibility to t. Hence in any subset $X \subseteq \mathbb{F}_{q^r}$ with $|X| > |T|$ there exists a value which preserves feasibility for every terminal simultaneously. Since $q > |T|$, subfield $\mathbb{F}_q$ is such a subset, which proves the theorem. □

Remark 4.1.11. Further advantage of our algorithm is that it alters the value of each local coefficient at most only once. In addition, there is no restriction on the order of determining the coefficients.

4.1.3 An application in wireless relay networks

In this subsection we describe the model in [3] approximating the capacity of a Gaussian wireless relay network, and explain why it is a special case of network code completion.

A deterministic wireless relay network consists of a source s, a terminal t and a set of transmitters L partitioned into ℓ layers $L = \{L_1 \cup \ldots \cup L_\ell\}$. (Note that this notion of layers is completely different from the concept of multi-layered video streams.) Each transmitter in $x \in L$ is represented by

two sets of nodes I_x and O_x, modelling the input and output of x, respectively. Connection between I_x and O_x is given by a directed bipartite graph $G_x(I_x, O_x, E_x)$, where I_x and O_x are the color classes of the bipartite graph and E_x is the set of arcs, each arc oriented from I_x to O_x. Let I_j denote the set of all input nodes of transmitters in layer j. Similarly we define O_j. The source s can only send messages to input nodes of the transmitters in the first layer L_1, that is, I_1. Output nodes in L_j can only transmit messages to input nodes in L_{j+1} and finally, terminal t receives information from output nodes of layer L_ℓ only. These connections are also modelled by directed bipartite graphs between $G_s(s, I_1, E_s)$, $G_j(O_j, I_{j+1})$ $(1 \leq j < \ell)$ and $G_t(O_\ell, t, E_t)$ with orientation from s to I_1 or from O_ℓ to t and from lower layer to higher. Similarly to network coding, messages sent through the network are members of a finite field $\mathbb{F}_q$ denoted by $M_1, \ldots, M_k$. In the unicast transmission k messages are sent from s to t. On each arc a linear combination of the messages is sent, which we will call **packets**.

Definition 4.1.12. A **transmission** of k messages in a deterministic wireless relay network consists of three components:

- a function $m : E_s \to \mathbb{F}_q^k$,
- a subset A of input and output nodes called **active** nodes, such that for each transmitter x, there exists a perfect matching between $I_x \cap A$ and $O_x \cap A$ in E_x,
- for each transmitter x, a perfect matching P_x connecting $I_x \cap A$ and $O_x \cap A$.

Given these components, packets sent on the arcs can be determined the following way. Function m describes packets sent on arcs of E_s: for an arc $e \in E_s$, packet $(M_1, \ldots, M_k) \cdot m(e)$ is sent. For an arc $o_{x_j} i_{x_{j+1}}$ between two layers, if both endnodes active, the packet sent is the same as the packet sent on the arc in P_{x_j} connecting o_{x_j} to its input pair. Finally, for an arc $i_x o_x$ connecting inputs and outputs in P_x, the packet sent is the sum over $\mathbb{F}_q$ of

the packets on arcs incident to i_x and connecting i_x with an active output node from the previous layer. A transmission sends k messages to t if taking the k-length coefficient vectors of the messages on packets of arcs in E_t, they span the whole k-dimensional space $\mathbb{F}_q^k$.

The **deterministic wireless relay network problem** is to decide whether a transmission sending k messages from s to t exists. The capacity of a deterministic wireless relay network is the maximum number of messages that can be sent with one transmission.

Several polynomial time algorithms were given for the unicast version of the deterministic wireless relay network problem. Yazdi and Savari [46] applied submodular flow techniques, Amaudruz and Fragouli [2] used augmenting paths, which Shi and Ramamoorthy [41] accelerated, and Goemans, Iwata and Zenklusen [18] solved the problem with matroid union or intersection. All of the approaches rely on the layered property of the model.

In [10], Fragouli and Ebrahimi, then Erez, Kim, Xu, Yeh, Medard in [23] showed that the NCCP has an application for deterministic wireless relay network models like the one defined above. Both also gave a min-max theorem for the network capacity [4], [23].

Theorem 4.1.1 (Kim, Médard [23])**.** *A deterministic wireless relay network problem can be modelled as a special case of a network code completion problem.*

Proof. Assume that a deterministic wireless relay network problem is given. First we define a corresponding digraph $D = (V, A)$. The node set contains s, t and all input and output nodes. Arcs are also copied to D with the same orientation. Finally, an extra node v_e is added for each arc $e = (i_x o_x)$ connecting an input and output of a transmitter x within a layer, subdividing the connecting arc. Some local coefficients are also fixed the following way. Local coefficients of the form $(v_e o_{x_j}, o_{x_j} i_{x_{j+1}})$ or $(o_{x_j} i_{x_{j+1}}, i_{x_j} v_e)$ are fixed to 1. Note that the only local coefficients not fixed are of the type $(i_{x_j} v_e, v_e o_{x_j})$. First we show that a transmission sending k messages can be transformed into a feasible network code completion on the corresponding digraph.

Lemma 4.1.13. *If there is a transmission sending k messages from s to t then there is a solution of the corresponding network code completion problem.*

Proof. If we set $\mathbf{c} := m$ on each outgoing arc of s and set a local coefficient to 1 if $(i_{x_j}v_e, v_e o_{x_j}) \in P_x$ then it is easy to check that the resulting network code defines the same sent packets, giving a feasible solution of the network code completion problem. □

Lemma 4.1.14. *If there is a solution of the corresponding network code completion problem, then there is a transmission sending k messages from s to t.*

Proof. Assume first that a solution of the network code completion problem is given with local and global coefficients α and $\mathbf{c}$, respectively. If such a solution exists, since $|T| = 1$, applying Remark 4.1.6 to set $\{0, 1\}$, we may assume that all non-fixed local coefficients are either 0 or 1. For every bipartite graph E_x, let F_x denote the set of those arcs for which the local coefficient corresponding to the subdividing node is 1. We show in two steps that this subset can be chosen to be a matching.

Claim 4.1.15. *If for a transmitter x a node $o_x \in O_x$ has degree $d > 1$ in F_x, then at least $d - 1$ local coefficients corresponding to entering arcs can be set to zero resulting in a feasible network code.*

Proof. Let $i_{x_1}, \dots i_{x_d}$ be the tails of arcs in F_x and let $\mathbf{z}_1, \dots, \mathbf{z}_d$ denote the global coefficients of arcs entering o_x. Let $\mathbf{c}_i$ denote the global coefficients of the network code attained from $\mathbf{c}$ by setting local coefficient $(i_{x_j}v_{e_j}, v_{e_j}o_x)$ to zero. It is enough to prove that at least one of $\mathbf{c}_1, \dots, \mathbf{c}_d$ is feasible, then we can decrease the degree of o_x in F_x one by one until $d = 1$ holds. Since all local coefficients with o_x as a middle node are fixed to one, if we apply Lemma 4.1.3 to $\mathbf{c}_i$ it is easy to see that all δ_a values are the same for each $1 \leq i \leq d$. Then we get that if all $\mathbf{c}_i$ global coefficients gave an infeasible network code, then so would $\mathbf{c}$, a contradiction. □

Claim 4.1.16. *If for a transmitter x a node $i_x \in I_x$ has degree $d > 1$ in F_x, then at least $d-1$ local coefficients corresponding to outgoing arcs can be set to zero resulting in a feasible network code.*

Proof. The statement can be proved similarly to the previous claim. □

Let us choose a feasible $0-1$ network code solution according to Claims 4.1.16, 4.1.15. We get that F_x is a matching for each x, and we choose $P_x = F_x$ and active node are the endpoints of these arcs. Let us define $\mathbf{c} = m$ on E_s. Then we get that packets sent on the arcs are exactly the ones determined by the network code also, which proves the lemma. □

Combining Lemmas 4.1.13 and 4.1.14 we get the proof of the theorem. □

Multicasting in deterministic wireless relay networks

The multicast version of the problem can be defined similarly to the unicast case, except that a set of terminals T is given and a transmission sends k messages if all terminals receive all messages. This problem can also be modelled with network code completion.

Considering multicast capacity, similarly to the case of the original network coding problem, nodes need to be able to perform network coding in order to achieve the maximal multicast capacity. Note that here coding only means that we let local coefficients of the form $(i_x v_e, v_e o_x)$ to have values different from 0 and 1, that is, we let $i_x o_x$ type arcs to multiply their transmitted packet by a constant from the finite field.

It has been shown in several independent papers that the multicast capacity with network coding equals the minimum of the unicast capacities: Kim and Médard gave a randomized [23] while Yazdi and Savari [45] and Ebrahimi and Fragouli [10] gave deterministic algorithms for the problem.

Former randomized algorithms for the NCCP have a lower bound on the required field size which depends on the size of the network and the number of terminals. We eliminated the first factor and presented randomized algorithms for both the unicast and multicast cases over any field of size greater than the

number of terminals. Our approach relies on the idea of a simple deterministic algorithm for the NCCP, constructing a solution for the multicast problem from solutions of the unicast special case.

4.2 Fixable sets and applications in heterogenous networks

First we define the problem of fixable pairs. Assume that a network coding problem is given.

Definition 4.2.1. We say that $M \subseteq L$ is **fixable** if **any** nonzero-valued mapping $\alpha_0 : M \to \mathbb{F}_q - \{0\}$ is extendable. The **fixable pairs problem** is to decide whether a given set M is fixable or not. For a pair $\ell = (wu, uv) \in L$, wu and uv are the **first** and **second** arcs of the pair, respectively, and u is the **central node** of the pair. Two pairs ℓ_1 and ℓ_2 are **consecutive** if the second arc of ℓ_1 is the first arc of ℓ_2. A path contains a pair, if it contains both of its arcs. For a subset of pairs $M \subseteq L$, a node is **M-influenced** if it is the central node of a pair in M. A set of paths is **M-independent** if they are pairwise arc-disjoint and any M-influenced node is contained by at most one of them.

4.2.1 Sufficient Condition for a fixable set

In this section we give a sufficient condition for a subset M of pairs to be fixable and present some applications in heterogeneous networks.

Theorem 4.2.2 (B-K, Király [26])**.** *Let $D = (V, A)$ an acyclic directed graph and $T \subseteq V - s$ a terminal set having a feasible network code for k messages over $\mathbb{F}_q$ with $q > |T|$, and let $M \subseteq L$ be a subset of pairs. If for every terminal $t \in T$ there exist k M-independent paths from s to t, such that none of the paths contains two consecutive pairs in M, then M is fixable.*

for $i = 1...m$ **do**
  if $\ell_i = (a, a')$, $a \in B$ and ℓ_i is contained in path P_j **then**
    $B \leftarrow B - a$
    $\text{arc}_{\text{new}} \leftarrow a'$
    if a' is the first arc of a pair $\ell' = (a', a'') \in M$ that is also contained in P_j **then**
      $\text{arc}_{\text{new}} \leftarrow a''$
    end if
    $B \leftarrow B + \text{arc}_{\text{new}}$
    if $\mathbf{c}(B)$ is not a basis then $\alpha(\ell_i) \leftarrow 1$.
  end if
end for

Figure 4.1: Exchanging an arc in B

Proof. We follow similar ideas for the network code construction as the ones in [21] but instead of determining the global coefficients one-by-one in a topological order we determine the local coefficients in a special order. Let α_0 be an arbitrary nonzero-valued mapping on M. From Theorem 4.1.2, M is extendable if and only if it is extendable for every one-element terminal set $\{t\}$. Let t be an arbitrary given terminal in T. We choose k M-independent st-paths $P_1, \ldots, P_k$ such that no path contains two consecutive pairs in M. (This can be done by applying classical graph transformation techniques.) First we consider the extension α of α_0 which is zero on $L \setminus M$. Let $\mathbf{c}$ denote the global coefficients corresponding to α. If α alters during the algorithm then $\mathbf{c}$ is modified accordingly. Let us fix a topological order of the nodes and let $\ell_1, \ldots, \ell_{|L \setminus M|}$ be an order of the pairs in $L \setminus M$ according to the topological order of the heads of the second arcs. We determine the values of α on the pairs in this order and maintain a set of arcs $B = \{b_1, \ldots, b_k\}$, such that $b_i \in P_i$, and $\mathbf{c}(B) = \{\mathbf{c}(b_i)\}_{i=1}^k$ forms a basis, and that B finally contains only arcs entering t. First, let $B = \{a_1, \ldots, a_k\}$. If there is a pair of the form (a_i, a) in M, where $a \in P_i$, then we replace a_i with a in B, see Figure 1.

Claim 4.2.3. *The modification of $\alpha(\ell_i)$ does not modify any arc in $B - \text{arc}_{\text{new}}$.*

Proof. The modification of $\alpha(\ell_i)$ influences an arc e if and only if there is a

path $a' = e_0, e_1, e_2, \ldots, e_z = e$ such that $\alpha(e_x, e_{x+1})$ is not zero for $0 \leq x < z$. Note that such a pair cannot be in $L \setminus M$ because these pairs have bigger index in our ordering, so their α-values would be still zero. Hence $(e_x, e_{x+1}) \in M$ for each $0 \leq x < z$. Suppose that there exists such an arc e which is in B and is not contained in P_j. Since P_j contains the head of a', which is the central node of (a', e_1) and the paths are M-independent, e_1 cannot be in a path different from P_j, so $z \geq 2$. Hence both the tail and head of e are different from the head of a'. There was a point when e got into B, let $\ell(e)$ be the pair that was being processed at that point. There are two cases: either e is the second arc of $\ell(e)$ or e is an arc following the second arc of $\ell(e)$. In both cases, the head of the second arc of $\ell(e)$ is reachable from the head of the second arc of ℓ_i. From the choice of the order of processing the pairs, $\ell(e)$ should be processed later than ℓ_i, which contradicts that $\ell(e)$ was processed earlier than ℓ_i. □

Claim 4.2.4. *After processing a pair,* $\mathbf{c}(B)$ *form a basis of* $\mathbb{F}_q^k$.

Proof. If $\alpha(\ell)$ remained 0, the claim clearly holds. Using Claim 4.2.3 we observe that we can apply Claim 1.2.7, so only one value β is wrong. Thus as zero was wrong, value 1 must be good. □

Claim 4.2.5. *After processing a pair* ℓ *in the algorithm, for every arc* b *in* B *one of the following hold:*

- b *enters* t,
- *for the arc* b' *following* b *on* P_j *the pair* (b, b') *is not in* M.

Proof. Suppose that arc b does not enter t and $(b, b') \in M$. Let us take the step when b got into B. From the choice of P_j, there are no consecutive pairs from M on P_j, so b cannot be the second arc of a pair in M contained in P_j. Hence arc_{new} should have been b' instead of b. □

From Claim 4.2.5 and the choice of the order of pairs the final set B will only contain arcs entering t, which proves the theorem. □

Remark 4.2.6. It remained open to give an exact characterization for a subset of pairs to be fixable.

4.2.2 Heterogeneous networks

Here we give a characterization for the network capacity including the case when some of the nodes are broadcasting. Let a network coding problem be given. Suppose that a subset $W \subseteq V \setminus (T \cup \{s\})$ of intermediate nodes can only **broadcast** messages, that is, such a node sends the same message on each of its outgoing arc. The **W-broadcasting network coding problem** is to decide the existence of a network code where every node in W broadcasts. To the best of our knowledge, there has been no characterization known on the existence of a feasible W-broadcasting network code. We say that a set of st-paths is **W-disjoint** if the paths are pairwise arc-disjoint and each node in W is contained in at most one of the paths. Note that the existence of k W-disjoint st-paths can be checked in polynomial time.

Theorem 4.2.7 (B-K, Király, [26])**.** *Given a W-broadcasting network coding problem with $q > |T|$ and , there exists a feasible network code, if and only if for every $t \in T$ there are k W-disjoint st-paths.*

Proof. We are going to reduce the W-broadcasting problem to a special case of a fixable pairs problem. Let D' denote the graph attained from D by expanding each node $w \in W$ into two new nodes w_i and w_o with a new arc $w_i w_o$ such that the incoming and outgoing arcs of w become the incoming and outgoing arcs of w_i and w_o, respectively. For a node $v \in V \setminus W$ let $v = v_i = v_o$. If there is a feasible W-broadcasting network code $\mathbf{c}$ on D, then it can be modified to be a feasible network code $\mathbf{c}'$ on D' by setting for $uv \in A: \quad \mathbf{c}'(u_o v_i) = \mathbf{c}(uv)$ and for $w \in W$ and for any $wv \in A$ we define $\mathbf{c}'(w_i w_o) = \mathbf{c}(wv)$, this is legal as w is a broadcasting node, consequently $\mathbf{c}(wv)$ is the same on every outgoing arc. Clearly, the existence of k W-disjoint st-paths in D is equivalent with the existence of k arc-disjoint st-paths in D'. This gives that the conditions of the theorem are necessary. For the other direction, let M be the following subset of pairs in D': $M = \{(w_i w_o,\ w_o v_i) \mid w \in W,\ wv \in A\}$. Note that M does not contain consecutive pairs and since every central node of a pair in M has in-degree one, M-

independentness follows from arc-disjointness. Applying Theorem 4.2.2 we get that if there exist W-disjoint paths in D then M is fixable in D' and so we can take a feasible extension of the constant 1-valued mapping on M. Let $\mathbf{c}'$ denote the global coefficients of the network code. One can easily get the global coefficients of a feasible network code on D by contracting arcs in D' of the form $w_i w_o,\ w \in W$. □

The fixable pairs problem can similarly model restrictions on incoming messages, if each node in a subset W can only receive a fixed nowhere zero linear combination of their incoming messages. This can also be handled applying Theorem 4.2.2 on the auxiliary graph D' of Theorem 4.2.7, by fixing for each $w \in W$ the local coefficient on a pair of the form $(u_o w_i, w_i w_o)$ to the corresponding value in the fixed linear combination.

Theorem 4.2.8 (B-K, Király [26])**.** *Let a network coding problem be given with $q > |T|$ and a subset $W \subseteq V$ such that every node in W only receives a fixed nowhere zero linear combination of its incoming messages. There exists a feasible network code if and only if for every $t \in T$ there are k W-disjoint st-paths.*

An important application is when some intermediate nodes only receive the XOR of their incoming messages (we assume that messages are from a finite field of size 2^d represented by d bits). The XOR of the incoming messages can be regarded as the sum over the finite filed, and for each $w \in W$, every local coefficient on a pair of the form $(u_o w_i, w_i w_o)$ is fixed to 1.

Chapter 5

Failure protecting network codes

In this chapter we present some applications of network coding for efficient failure protection [8]. Our goal is to find a network code that resists a certain number of arc failures, that is, deleting any subset of the arcs the network code remains sufficient on the remaining subgraph without altering the local coefficients. Such a network code requires minimal occupation of the network but enables instant recovery. In the first part of this chapter we show efficient algorithms and lower field size bounds for network code construction. In the last section we present some negative results for the capacity case of the problem [5].

5.1 Introduction

In every algorithm for network code construction the required field size is an important parameter, because efficient algorithms require small field sizes. As an example, for the classical multicast linear network code construction in acyclic graphs, Li, Yeung and Cai in [32] showed that the max flow-min cut property is necessary and sufficient for the existence, but their lower bound on a sufficient field size depends on the size of the graph. It was Koetter and Médard in [27] who showed that actually a lower bound of $O(|T|k)$ is enough. With other words, the required field size does not depend on the size of the graph. Later Jaggi et al. [21] improved this lower bound to $|T|$. Our result can be regarded as a step in an analogue series of results for failure protecting network codes. Harvey et al. showed that for the existence of failure protecting

network codes a natural min-max type condition is not only necessary but also sufficient [19]. In their approach, the lower bound for the required field size depends on the size of the graph. In this chapter we show that a lower bound on the field size can be given which is independent from the number of nodes or arcs in the graph. Our proofs rely on the connection with the topic of network encoding complexity. Ideas used in the proofs are basically similar to the techniques used in [13] for achieving better field size bounds in wiretap networks.

5.2 Problem formulation

The notion of failure protection can be defined in several ways. We use the definition of [19] for failure protecting network codes, and our goal is to find a network code that remains feasible after a certain number of arc failures , that is, deleting any subset of the arcs the network code remains feasible on the remaining subgraph without altering local coefficients on failureless pairs.

Definition 5.2.1. Assume a network code $(\alpha, \mathbf{c})$ is given on a network. An **failure** is a subset of arcs $H \subseteq A$, and the network code $(\alpha_H, \mathbf{c}_H)$ resulting from $(\alpha, \mathbf{c})$ by H is setting all local coefficients to zero on pairs intersecting H (a pair (uv, vw) intersects H if $\{uv, vw\} \cap H \neq \emptyset$). If $(\alpha, \mathbf{c})$ is a feasible network code, we say that $(\alpha, \mathbf{c})$ **protects failure** H, if $(\alpha_H, \mathbf{c}_H)$ is also feasible. For a positive integer $d \in \mathbb{N}$ a network code is **d-failure-protecting**, if it protects any failure of size at most d. Similarly, given a set $\mathcal{H} \subseteq 2^A$ of possible failures, a network code is **$\mathcal{H}$-protecting** if it protects every failure $H \in \mathcal{H}$.

Note that such network codes enable a very fast, in fact instant recovery, because there is no need to inform the whole network about the failure, only a node with a failing entering arc needs to be able to recognize the failure.

5.3 Previous and new bounds

5.3.1 Related work

In [19], the existence of a protecting code over sufficiently large fields was characterized and a polynomial time algorithm was given by reducing the problem to simultaneous matrix completion. Given a failure H, it is easy to see that for the existence of an H-protecting network code it is necessary that there remain k arc-disjoint paths to every receiver node from s in $(V, A \setminus H)$. In [19], Harvey et al. showed that this is also sufficient, even for failure sets.

Theorem 5.3.1 (Harvey, Karger, Murota [19]). *There exists an $\mathcal{H}$-protecting network code (α, c) if and only if for every $H \in \mathcal{H}$ there are k arc-disjoint paths from s to every receiver in $(V, A \setminus H)$. Moreover, a protecting network code can be chosen over any field of size $q > |T||\mathcal{H}|$ in time $O(|T||\mathcal{H}|(m^3 \log m + |L|m^2))$.*

We can deduce the following theorem for the special case of d-protection.

Theorem 5.3.2 (Harvey, Karger, Murota [19]). *There exists a d-failure protecting code if and only if $\lambda(s,t) \geq k + d$ for every receiver t. Such a code can be found over any finite field of size at least $|T|(\binom{m}{d} + \cdots + \binom{m}{0})$ in time $O(|T|(\binom{m}{d} + \cdots + \binom{m}{0})(m^3 \log m + |L|m^2))$.*

5.3.2 New bound

The main result of this chapter is that the term m can be eliminated from sufficient field size bound for d-protection. Bahramgiri and Lahouti in [7] also gave similarly network size independent lower bounds for the required field size, but their algorithm used random network codes. For a summary on other related results see the survey of Sanna and Izquierdo [39].

Theorem 5.3.3 (B-K [8]). *If a d-failure protecting network code exists, then such a network code can be found over any field of size $q > |T|(\binom{N}{d} + \cdots +$*

$\binom{N}{0}$), *where* $N = 3\binom{|T|}{2}(k+d)^3$. *The running time of such an algorithm is* $O(m^2|T|(k+d) + |T|(\binom{N}{d} + \cdots + \binom{N}{0})N^3 \log N)$.

This lower bound on the field size may be much smaller for large graphs. The idea of the proof is based on a completely different topic called network encoding complexity. The proof will be described in 5.4.3, but first some results from the area of encoding complexity are presented, which we will use for the proof.

5.4 Network encoding complexity

5.4.1 Definitions and previous results

When constructing network codes in practice, one may notice that typically very few nodes perform actual coding, most nodes just forward one of the incoming messages on each outgoing arc.

Definition 5.4.1. Given a network code $(\alpha, \mathbf{c})$, a node v is called a **coding node** if there exists an outgoing arc vz such that there are at least two non-zero local coefficients $\alpha(uv, vz)$ and $\alpha(wv, vz)$ corresponding to vz.

In the butterfly network for example (Figure 1.1), node w is the only coding node. The topic of network encoding complexity deals with the following fundamental question: Given a network coding problem, what is the minimum number of coding nodes for a feasible network code construction?

In [30], it was proved that the number of coding nodes can always be bounded by a value independent of the size of the network.

Theorem 5.4.1 (Langberg, Sprintson, Bruck, [30])**.** *Given a network such that* $\lambda(s,t) \geq k$ *for every* $t \in T$*, there exists a feasible network code over any finite field* $\mathbb{F}_q$ *with* $q > |T|$ *with at most* $k^3\binom{|T|}{2}$ *coding nodes.*

The proof of the theorem is based on a reduction of the original graph to an auxiliary graph with all internal nodes of degree two or three. Since we

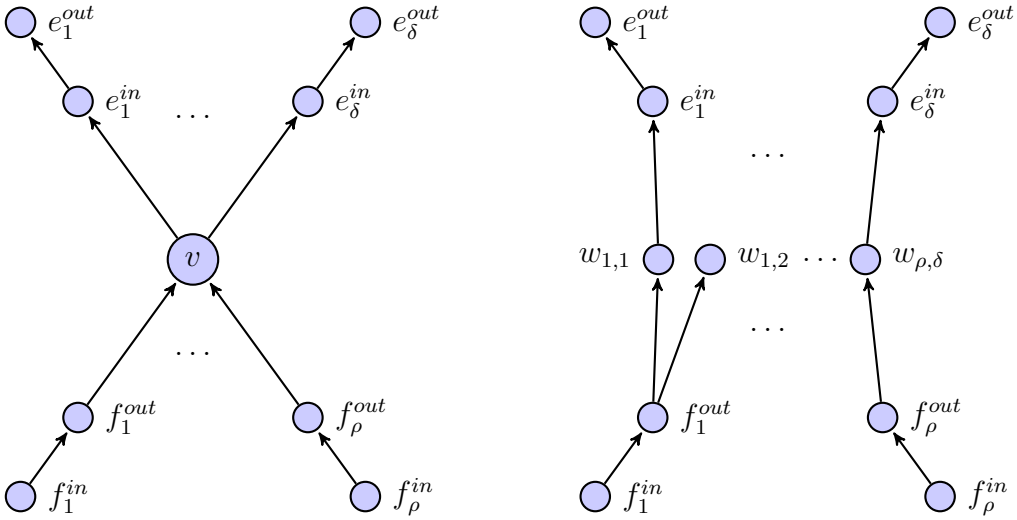

Figure 5.1: Substitution of node v in the first step.

also use this auxiliary graph in our proof, we repeat its detailed construction in the following subsection.

5.4.2 Auxiliary graph construction

We describe a graph construction slightly different from that in [30]. We can assume that each receiver node is a sink. The construction has four steps. In the first step, each arc e is subdivided by two new nodes e^{in} and e^{out}. Then each internal node is substituted by a set of arcs as follows. Let v denote such a node with in-degree ρ and out-degree δ and with incoming and outgoing arcs $f_1, f_2, \ldots, f_\rho$ and $e_1, \ldots, e_\delta$, respectively. For each $1 \leq i \leq \rho$ and $1 \leq j \leq \delta$ we add node $w_{i,j}$ and arcs $(f_i^{out}, w_{i,j})$ $(w_{i,j}, e_j^{in})$ to the graph (see Fig. 5.1). Note that in this new graph nodes of the form e^{in} and e^{out} have out-degree and in-degree one, respectively. In the second step, if a node of the form e_j^{in} has in-degree more than two, the incoming arcs are substituted by a graph, where each node has in-degree at most two. Let $w_{1,j}, w_{2,j}, \ldots, w_{\rho,j}$ denote the tails of arcs entering e_j^{in}. Then for each $2 \leq i \leq \rho - 1$, arc $w_{i,j}$ is subdivided by a new node z_i and the head of arc $(w_{i+1,j}, e_j^{in})$ is replaced by z_i, as shown in Figure 5.2. Similar procedure is made on nodes of the form e^{out} with out-degree more

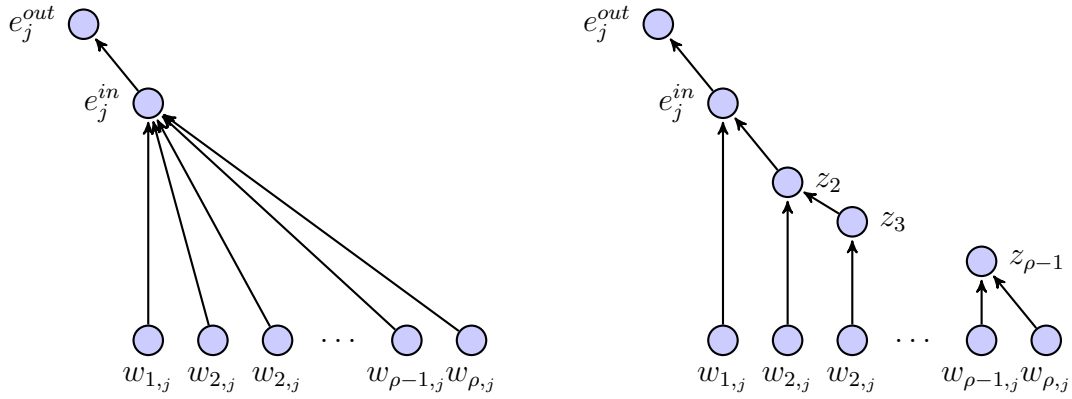

Figure 5.2: Substitution of arcs entering e_j^{in}.

than two.

The resulting graph after the two steps is denoted by $D' = (V', A')$. Note that every internal node in D' has total degree at most three. We can observe the following relationship between paths in D and D'.

Proposition 5.4.2. *Let (f_i, e_j) be a pair in D. Then there is exactly one path P_i^j from f_i^{out} to e_j^{in} in D'. P_i^j is edge-disjoint from path P_k^l if and only if $i \neq k$ and $j \neq l$.*

In the third step, we omit arcs from D' as long as the connectivity requirements between s and the receivers are satisfied ($\lambda(s,t) \geq k$ for all $t \in T$). Let us denote the remaining graph $D'' = (V'', A'')$.

Finally, in the fourth step nodes with exactly one indegree and outdegree are replaced by a single arc as follows.

Definition 5.4.3. A **branch** of a graph D is a directed uv-path P in D such that $\rho(u) \neq 1$ and $\delta(v) \neq 1$ but all other in-degrees and out-degrees of nodes in P are one.

Note that every arc in a graph is covered by exactly one branch. Specifically, in D'' only nodes in $T + s$ or with total degree three can be endpoints of a branch. Let us substitute each branch of D'' by a single arc and let us denote the final graph by $D^* = (V^*, A^*)$. In [30], Theorem 5.4.1 was proved by the following lemma.

Lemma 5.4.4. *D^* has at most $\binom{|T|}{2}k^3$ internal nodes.*

For our proof we need to define a mapping ϕ from A'' to A^*: for an arc $e \in A''$, $\phi(e)$ is the arc corresponding to the branch containing arc e in D''. For a set of arcs $H \subseteq A''$, by abuse of notation, let $\phi(H) := \{\phi(e)|e \in H\}$.

5.4.3 Proof of Theorem 5.3.3

From Theorem 5.3.1 we get that for the existence of a d-protecting network code on D, the existence of $k+d$ arc-disjoint paths from s to each receiver is a necessary and sufficient condition. So we can construct the auxiliary graph $D^* = (V^*, A^*)$ for $k+d$ paths. The idea is to find a d-protecting network code on D^* over a finite field, finally map it to an failure protecting code on D over the same field.

From Theorem 5.4.1 we get that $|A^*| \leq 3\binom{|T|}{2}(k+d)^3$. Since internal nodes have degree three, $|L| \leq 4|A^*|$, so setting $N = 3\binom{|T|}{2}(k+d)^3$ we get that $O(m^3 \log m + |L|m^2) \leq O(N^3 \log N + N^3) = O(N^3 \log N)$. Note that d-protection is equivalent to $\mathcal{H}$-protection, where $\mathcal{H}$ is the set containing all subsets of A^* of size at most d. Since $|\mathcal{H}| = \binom{|A^*|}{d} + \ldots + \binom{|A^*|}{0}$, by applying Theorem 5.3.1 to D^*, we get that there exists a d-protecting network code $(\alpha^*, \mathbf{c}^*)$ on D^* over any finite field of size at least $|T||\mathcal{H}|$, where $|\mathcal{H}|$ is a function of $|T|, k, d$.

Lemma 5.4.5. *A d-protecting network code $(\alpha^*, \mathbf{c}^*)$ on D^* can be transformed into a d-protecting network code $(\alpha, \mathbf{c})$ on D over the same finite field.*

For continuity we leave the proof of the lemma for the next subsection. Applying Lemma 5.4.5 for network code $(\alpha^*, \mathbf{c}^*)$ we get Theorem 5.3.3.

5.4.4 Proof of Lemma 5.4.5

We map $(\alpha^*, \mathbf{c}^*)$ one-by-one to D'' then D' and finally D, always maintaining d-protection and the field size. The existences of these mappings are proved by three claims. First we show that a network code on D' can be

naturally transformed into one on D , and failure protection is preserved on arcs of type (e^{in}, e^{out}). For an arc set $F \subseteq A$, let us define $F' := \{(f^{in}, f^{out}) \in A' | f \in F\}$. Similarly, we define $F'' := F' \cap A''$.

Claim 5.4.6. *For a network code $(\alpha', \mathbf{c}')$ on D', there exists a network code $(\alpha, \mathbf{c})$ on D over the same field such that for every failure set F, if $(\alpha', \mathbf{c}')$ is F'-protecting then $(\alpha, \mathbf{c})$ is F-protecting.*

Proof. Let $\mathbf{c}(e) := \mathbf{c}'(e^{in}, e^{out})$. Observe from Figure 5.1 that arcs (f_1^{in}, f_1^{out}), $\ldots, (f_\rho^{in}, f_\rho^{out})$ form an s, e_j^{in}-cut in D'. Hence we have that $\mathbf{c}'(e_j^{in}, e_j^{out}) \in \langle\{\mathbf{c}'(f_i^{in}, f_i^{out}) | 1 \leq i \leq \rho\}\rangle$. If we set a local coefficient $\alpha(f, e)$ as the product of local coefficients of α' along path P_i^j, then $\mathbf{c}_F(e) = \mathbf{c}'_{F'}(e_j^{in}, e_j^{out})$ for every arc $e \in A \setminus F$ and the failure-protecting part of the claim follows. □

The following claim shows the relation between network codes on D'' and D'.

Claim 5.4.7. *A network code $(\alpha'', \mathbf{c}'')$ on D'' corresponds to a network code $(\alpha', \mathbf{c}')$ on D' such that for every failure $M' \subseteq A'$, if $(\alpha'', \mathbf{c}'')$ is M''-protecting then $(\alpha', \mathbf{c}')$ is M'-protecting.*

Proof. A network code on D'' can be regarded as a network code on D' such that $\mathbf{c}(e) = \mathbf{c}'(e)$ for every arc $e \in A''$ and $\alpha'(e, f) = \alpha(e, f)$ if $e, f \in A''$, whereas $\mathbf{c}'(e) = 0$ if $e \notin A''$ and α' set to zero on all pairs intersecting deleted arcs. □

Claim 5.4.8. *If there exists a d-failure protecting network code $(\alpha^*, \mathbf{c}^*)$ on D^*, then there exists a d-failure protecting network code $(\alpha'', \mathbf{c}'')$ on D'' over the same field.*

Proof. We set $\mathbf{c}''(e) := \mathbf{c}^*(\phi(e))$, and set all local coefficients on pairs of a branch of D'' to one. If two arcs e, f in A'' are on the same branch B, then $\mathbf{c}''_e = \mathbf{c}''_f$, and their failure corresponds to the same failure $\phi(e) = \phi(f)$ in A^*. So for every failure K in D'', $\mathbf{c}''_K = \mathbf{c}^*_{\phi(K)}$ hence the failure protection follows. □

Combining Claims 5.4.6, 5.4.7, 5.4.8 we get the proof of Lemma 5.4.5.

5.5 Bounds on unicast connections with capacities

This section considers slightly different setup for failure protecting network code construction. Instead of a set of terminals only a single terminal node, with other words, a unicast connection is considered, but arcs may have different capacities.

The first, surprising result in this scenario was published by Rouayheb, Sprintson and Georghiades in [14]. They showed that if two messages are sent from s to t in a digraph ($k = 2$) with arc capacities one or two such that there remains an st-flow of value two after the failure of any arc, then the two-element finite field $\mathbb{F}_2$ is always satisfactory for failure protecting network code construction. Their proof is based on a structural study of digraphs minimal to this property.

Later Babarczi, Tapolcai, Rónyai and Médard in [6] even further strengthened this observation by proving that encoding is actually only needed at the source node. Their key idea is to show that the network can be decomposed into three subnetworks such that the failure of any arc leaves at least two of them st-connected.

In this subsection we examine whether this nice decomposable property can be generalized for less special scenarios. Unfortunately, as it turns out, the slightest increase on the number of messages or possible capacities ruins the property. These results are presented in [5].

Two straightforward generalizations may be to increase the number of data flows or the number of possible link failures. In this section we present examples to show that none of these generalizations are possible, i.e., such flow decomposition algorithm may not exit for these scenarios.

5.5.1 Dual link failure resilience with two data parts

Figure 5.3 presents a network, where two data parts have to be sent from s to t. The connection can resist two failures, because after deleting any two edges of G, there are two remaining paths in G^* from s to t. Note that the graph is critical respect to this property.

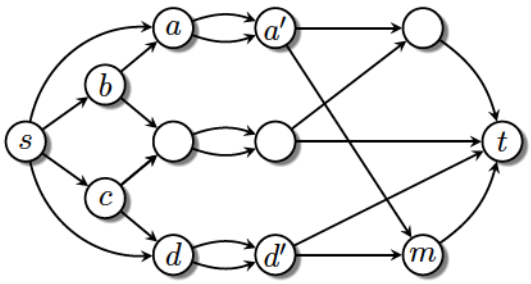

Figure 5.3: A network with two possible failures and two data parts where 4 end-to-end edge sets do not exist. Parallel arcs represent 2-capacity edges.

Edges $(s,a),(s,b),(s,c),(s,d)$ form a 4-edge-cut, so any two of them can remain after the failure of the other two. This shows that 3 end-to-end subgraphs do not guarantee protection against two failures, so 4 parts are needed, and each of these parts transmits a message such that the original data is decodable when receiving at least two of them. We show that edges of the network in Figure 5.3 cannot be split into 4 end-to-end subgraphs. Assume indirectly that they can, and let E_1, E_2, E_3, E_4 denote the sets of edges. Edges in a 4-edge-cut have to belong to separate subgraphs, hence it is easy to see that edges entering node a belong to different edge sets, and the same holds for d. Without loss of generality we may assume that edge $(a',m),(m,t) \in E_1$ and $(d',m) \in E_2$. Then edge (d',m) does not have an end-to-end connection in E_2, contradicting the assumption.

5.5.2 Single link failure with three or more data parts

Consider now the case when the original message is divided into three data parts (A, B and C) and at most one link can fail in the network. For example, in Figure 5.4, the failure of any link preserves a flow of value 3 from s to t. Edges $(s,a),(s,b),(s,c),(s,d)$ form a 4-edge-cut, so with the decomposition approach again 4 end-to-end sets of edges are needed to resist at most two edge failures in G^* (corresponding to a single failure of a capacity 2 edge in G). We show that such partition of the edges does not exists. Suppose indirectly that there are such sets (E_1, E_2, E_3, E_4) in the graph and assume that $(s,c) \in E_1$. Then so is (c,w) and (c,m). Since $(w,t),(m,t),(a',t),(d',t)$ also form a 4-edge-cut, exactly one of them belongs to E_1, and it is either (m,t) or (w,t). Because of symmetry we can assume it is (m,t) and $(w,t) \in E_2$. Then after the failure of link (a,a'), only two data parts can be transmitted to t, even if node w switches to (c,w), which clearly requires control plane signaling.

Note that by adding k further edges from s to t, the same argument holds for $k+3$ data parts.

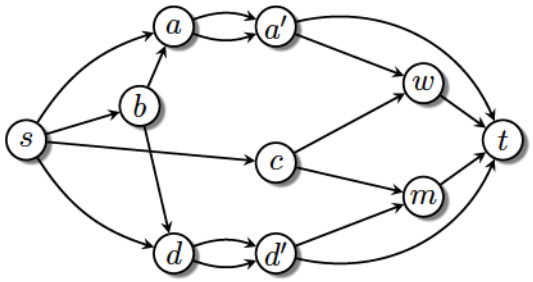

Figure 5.4: A network with one possible failure and three data parts where 4 end-to-end edge sets do not exist. Parallel arcs represent 2-capacity edges.

Bibliography

[1] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216, 2000. 2, 3, 5

[2] A. Amaudruz and C. Fragouli. Combinatorial algorithms for wireless information flow. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '09, pages 555–564, 2009. 65

[3] A. S. Avestimehr, S. N. Diggavi, and D. N. C. Tse. A deterministic approach to wireless relay networks. In *Proceedings of the 45th annual Allerton conference on Communication, control, and computing*, Allerton'07, 2007. 10, 62, 63

[4] A. S. Avestimehr, S. N. Diggavi, and D. N. C. Tse. Wireless network information flow. In *Proceedings of the 45th annual Allerton conference on Communication, control, and computing*, Allerton'07, 2007. 65

[5] P. Babarczi, J. Tapolcai, A. Pašić, L. Rónyai, E. R. Kovács, and M. Médard. Linear time coding algorithms for resilient flow decomposition in transport networks. *Preprint*, 2014. 11, 73, 81

[6] P. Babarczi, J. Tapolcai, L. Rónyai, and M. Médard. Resilient flow decomposition of unicast connections with network coding. In *2014 IEEE International Symposium on Information Theory (ISIT)*, pages 116–120, 2014. 81

[7] H. Bahramgiri and F. Lahouti. Robust network coding against path failures. *Communications, IET*, 4(3):272–284, 2010. 75

[8] E. R. Bérczi-Kovács. Graph independent field size bounds on failure protecting network codes. Technical Report TR-2015-01, Egerváry Research Group, Budapest, 2015. `www.cs.elte.hu/egres`. 11, 73, 75

[9] B. Dezső, A. Jüttner, and P. Kovács. Lemon–an open source C++ graph template library. *Electronic Notes in Theoretical Computer Science*, 264(5):23–45, 2011. 33

[10] J. Ebrahimi and C. Fragouli. Multicasting algorithms for deterministic networks. In *Information Theory Workshop (ITW)*, pages 1–5. IEEE, 2010. 65, 67

[11] J. Edmonds. Edge-disjoint branchings. *Combinatorial Algorithms*, 9:91–96, 1973. 2

[12] M. Effros. Universal multiresolution source codes. *IEEE Transactions on Information Theory*, 47(6):2113–2129, 2001. 13

[13] S. El Rouayheb, E. Soljanin, and A. Sprintson. Secure network coding for wiretap networks of type ii. *IEEE Transactions on Information Theory*, 58(3):1361–1371, 2012. 74

[14] S. El Rouayheb, A. Sprintson, and C. Georghiades. Robust network codes for unicast connections: A case study. *IEEE/ACM Transactions on Networking (TON)*, 19(3):644–656, 2011. 81

[15] E. Erez and M. Feder. Efficient network code design for cyclic networks. *IEEE Transactions on Information Theory*, 56(8):3862–3878, 2010. 62

[16] Erika R. Kovács and Morten V. Pedersen and Daniel E. Lucani and Frank H. P. Fitzek. Layer algorithms and simulation for the Kodo library, 2014. 48, 55

[17] M. Fresia, O. Y. Bursalioglu, G. Caire, and H. V. Poor. Multicasting of digital images over erasure broadcast channels using rateless codes. In *Sarnoff Symposium, 2009. SARNOFF'09. IEEE*, pages 1–6, 2009. 44

[18] M. X. Goemans, S. Iwata, and R. Zenklusen. An algorithmic framework for wireless information flow. In *Proceedings of the 47th annual Allerton conference on Communication, control, and computing*, Allerton'09, pages 294–300, 2009. 65

[19] N. J. A. Harvey, D. R. Karger, and K. Murota. Deterministic network coding by matrix completion. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 489–498, 2005. 59, 60, 74, 75

[20] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong. A random linear network coding approach to multicast. *IEEE Transactions on Information Theory*, 52(10):4413–4430, 2006. 62

[21] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. M. G. M. Tolhuizen. Polynomial time algorithms for multicast network code construction. *IEEE Transactions on Information Theory*, 51(6):1973–1982, 2005. 5, 7, 8, 19, 20, 21, 24, 69, 73

[22] M. Kim, D. Lucani, X. Shi, F. Zhao, and M. Médard. Network coding for multi-resolution multicast. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9, 2010. 9, 13, 14, 18, 31, 33, 40, 43, 44

[23] M. Kim and M. Médard. Algebraic network coding approach to deterministic wireless relay networks. In *Proceedings of the 48th Annual Allerton Conference on Communication, Control, and Computing*, pages 1518–1525, 2010. 60, 62, 65, 67

[24] Z. Király and E. R. Kovács. A network coding algorithm for multi-layered video streaming. In *Network Coding (NetCod), 2011 International Symposium on*, pages 1–7, 2011. 9, 15, 17, 20, 22, 24, 26, 28, 30, 43

[25] Z. Király and E. R. Kovács. Deterministic network coding algorithms and applications for wireless networks. In *Network Coding (NetCod), 2012 International Symposium on*, pages 103–107, 2012. 10, 60, 61, 63

[26] Z. Király and E. R. Kovács. Randomized and deterministic algorithms for network coding problems in wireless networks. *Information Processing Letters*, 115(4):507–511, 2015. 10, 68, 71, 72

[27] R. Koetter and M. Médard. An algebraic approach to network coding. *IEEE/ACM Transactions on Networking*, 11(5):782–795, 2003. 5, 73

[28] E. R. Kovács, M. V. Pedersen, D. E. Lucani Roetter, and F. Fitzek. Receiver heterogeneity helps: Network coding for wireless multi-layer multicast. *International Symposium on Network Coding*, pages 1–6, 2014. 9, 43

[29] J. Krigslund, F. Fitzek, and M. V. Pedersen. On the combination of multi-layer source coding and network coding for wireless networks. In *2013 IEEE 18th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pages 1–6, 2013. 43, 44

[30] M. Langberg, A. Sprintson, and J. Bruck. The encoding complexity of network coding. *IEEE/ACM Transactions on Networking (TON)*, 14(SI):2386–2397, 2006. 76, 77, 78

[31] A. R. Lehman and E. Lehman. Complexity classification of network information flow problems. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '04, pages 142–150, 2004. 15

[32] S.-Y. Li, R. W. Yeung, and N. Cai. Linear network coding. *IEEE Transactions on Information Theory*, 49(2):371–381, 2003. 5, 73

[33] Y. Li and E. Soljanin. Rateless codes for single-server streaming to diverse users. In *Proceedings of the 47th Annual Allerton Conference on Communication, Control, and Computing*, pages 1419–1426, 2009. 44

[34] Z. Li and B. Li. Network coding: The case of multiple unicast sessions. In *Allerton Conference on Communications*, volume 16, 2004. 8

[35] Z. Li, B. Li, and L. C. Lau. A constant bound on throughput improvement of multicast network coding in undirected networks. *IEEE Transactions on Information Theory*, 55(3):1016–1026, 2009. 8

[36] J. Liu, D. Goeckel, and D. Towsley. Bounds on the gain of network coding and broadcasting in wireless networks. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 724–732. IEEE, 2007. 8

[37] S. Maheshwar, Z. Li, and B. Li. Bounding the coding advantage of combination network coding in undirected networks. *IEEE Transactions on Information Theory*, 58(2):570–584, 2012. 8

[38] M. V. Pedersen, J. Heide, and F. H. P. Fitzek. Kodo: An open and research oriented network coding library. *NETWORKING 2011 Workshops*, pages 145–152, 2011. 48

[39] M. Sanna and E. Izquierdo. A survey of linear network coding and network error correction code constructions and algorithms. *International Journal of Digital Multimedia Broadcasting*, 2011, 2011. 75

[40] N. Shacham. Multipoint communication by hierarchically encoded data. In *INFOCOM'92. Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE*, pages 2107–2114, 1992. 13

[41] C. Shi and A. Ramamoorthy. Fast algorithm for finding unicast capacity of linear deterministic wireless relay networks. *arXiv preprint arXiv:0909.5507*, 2009. 65

[42] D. Vukobratovic and V. Stankovic. Unequal error protection random linear coding strategies for erasure channels. *IEEE Transactions on Communications*, 60(5):1243–1252, 2012. 44

[43] T. Wang, M. Médard, and L. Zheng. Joint design of multi-resolution codes and intra/inter-layer network coding. In *Conference Record of the*

46th Asilomar Conference on Signals, Systems and Computers (ASILOMAR), pages 1903–1907, 2012. 44

[44] J. Widmer, A. Capalbo, A. Anta, and A. Banchs. Rate allocation for layered multicast streaming with inter-layer network coding. In *INFOCOM, 2012 Proceedings IEEE*, pages 2796–2800, 2012. 18, 43, 44

[45] S. M. S. Yazdi and S. A. Savari. A deterministic polynomialtime algorithm for constructing a multicast coding scheme for linear deterministic relay networks. In *45th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6, 2011. 67

[46] S. M. S. T. Yazdi and S. A. Savari. A max-flow/min-cut algorithm for a class of wireless networks. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 1209–1226, 2010. 65

[47] R. W. Yeung, S.-Y. R. Li, N. Cai, and Z. Zhang. *Network Coding Theory - Part I: Single Source*, volume 2 of *Foundations and Trends in Communications and Information Theory*. Now Publishers Inc., 2005. 1

Summary

We considered different generalizations of the classical network coding min-max theorem and algorithm, with a focus on practical applications.

In Chapter 2, we investigated the multi-layered multicasting problem. We proved NP-hardness for some very special cases of the problem, including demand $\tau = (T_1, T_2)$, if we want to maximize the number of satisfied receivers. For two layers we gave a network coding algorithm which is optimal if the task is to send the base layer to every receiver and two layers to as many receivers as possible. For three or more layers we gave a sufficient condition for a feasible height function, and showed that this condition can be checked algorithmically. Also, we presented and implemented a heuristic for three layers, which sends the base layer to all receivers, and also carries the second layer to the maximum number of receivers. We gave an algorithm for the evaluation of some randomized network coding heuristics, and, as a further application, derived lower bounds on their performance.

In Chapter 3, we presented a novel code structure to manage multi-layer, multicast streams in practical wireless systems, taking user diversity into account. Permitting in the scheme on-demand, partial decoding of the refinement layer to obtain the base layer is unique, thus widening the array of tuning parameters, and favouring users with low demand but high computational capacity. Our estimations and implementation results both show that the coding structure reduces the time to complete the transmission.

In Chapter 4, we presented a new, simple, algorithmic proof for the min-max-type property of the multicast network code completion problem. We also gave randomized algorithm over any field $\mathbb{F}_q$ with $q > |T|$ for both the unicast and multicast cases. We proposed the fixable pair problem, gave a sufficient condition and showed some applications to networks with varying node transmission properties, giving a sharp characterization for the solvability of these problems.

In Chapter 5, we gave a lower bound of $|T|\left(\binom{N}{d} + \ldots + \binom{N}{0}\right), N = 3\binom{T}{2}(k+d)^3$ on the required field size for a d-protecting network code, and an algorithm for such a code construction with running time of $O(m^2|T|(k+d) + |T|(\binom{N}{d} + \cdots + \binom{N}{0})N^3 \log N)$. The importance of this bound is its independence of the network size. Also, we presented some negative results for the possible generalization for the capacitated case of the problem.

Összefoglalás

A tézis a hálózati kódolási alapfeladatra vonatkozó min-max tétel és algoritmus lehetséges általánosításait vizsgálja, elsősorban gyakorlati alkalmazásokból származó kérdéseken.

A második fejezetben a többrétegű többesküldési feladatot vizsgáltuk. Beláttuk a probléma NP-nehézségét több speciális esetre, például a teljesített igények maximalizálására két réteg esetén. Ugyancsak két réteg esetén viszont optimális algoritmust adtunk arra a feladatra, amikor az első réteget minden vevőcsúcsnak el kell küldeni, és cél a második réteg eljuttatása minél több vevőhöz.

Három vagy több réteg esetén algoritmikusan ellenőrizhető elégséges feltételt adtunk egy magasságfüggvény megengedettségére. Három réteg esetén javasoltunk és implementáltunk egy heurisztikát, mely minden vevőcsúcsnak elküldi az első réteget, és emellett a lehető legtöbb vevőhöz juttatja el a másodikat.

Determinisztikus algoritmust adtunk magasságkorlátozáson alapuló véletlen hálózati kódolási algoritmusok várható teljesítményének kiértékelésére, melyből egyúttal az ismert alsó becslésre is új bizonyítás adódott.

A harmadik fejezetben szintén egy többrétegű feladatra mutattunk be kódolási eljárást, de vezetéknélküli hálózatok esetén, a felhasználók eltérő számítási- és felbontási képességeit is figyelembe véve. Bevezettünk egy új paramétert a felsőbb rétegek részleges visszakódolására, mely csökkenti a nagy számítási kapacitással, de alacsony felbontású képernyővel rendelkező vevők várható letöltési idejét. Az implementációk és számítások alapján az átlag átviteli idő ezáltal csökkenthető.

A negyedik fejezetben egy új, egyszerű, algoritmikus bizonyítást adtunk a hálózati kód kiegészítési problémához kapcsolódó min-max tételre. Ennek segítségével véletlen algoritmusokat adtunk mind az egyes-, mind a többesküldési esetre, melyek tetszőleges, terminálszámnál nagyobb test esetén alkalmazhatók. Definiáltuk a rögzíthető párok problémáját, melyre elégséges feltételt adtunk. A kapott eredményeket néhány, vegyes tulajdonságú hálózaton definiált problémára alkalmaztuk, karakterizálva azok megoldhatóságát.

Az ötödik fejezetben $|T|\left(\binom{N}{d}+\ldots+\binom{N}{0}\right), N=3\binom{T}{2}(k+d)^3$ értékű alsó korlátot és $O(m^2|T|(k+d)+|T|(\binom{N}{d}+\cdots+\binom{N}{0})N^3\log N)$ futási idejű algoritmust adtunk az elégséges testméretre d-védő hálózati kód konstruálásához. A korlát lényeges tulajdonsága a függetlenség a hálózat méretétől. Végül bemutattunk néhány negatív eredményt a kapacitásos esetre vonatkozó tétel általánosításairól.